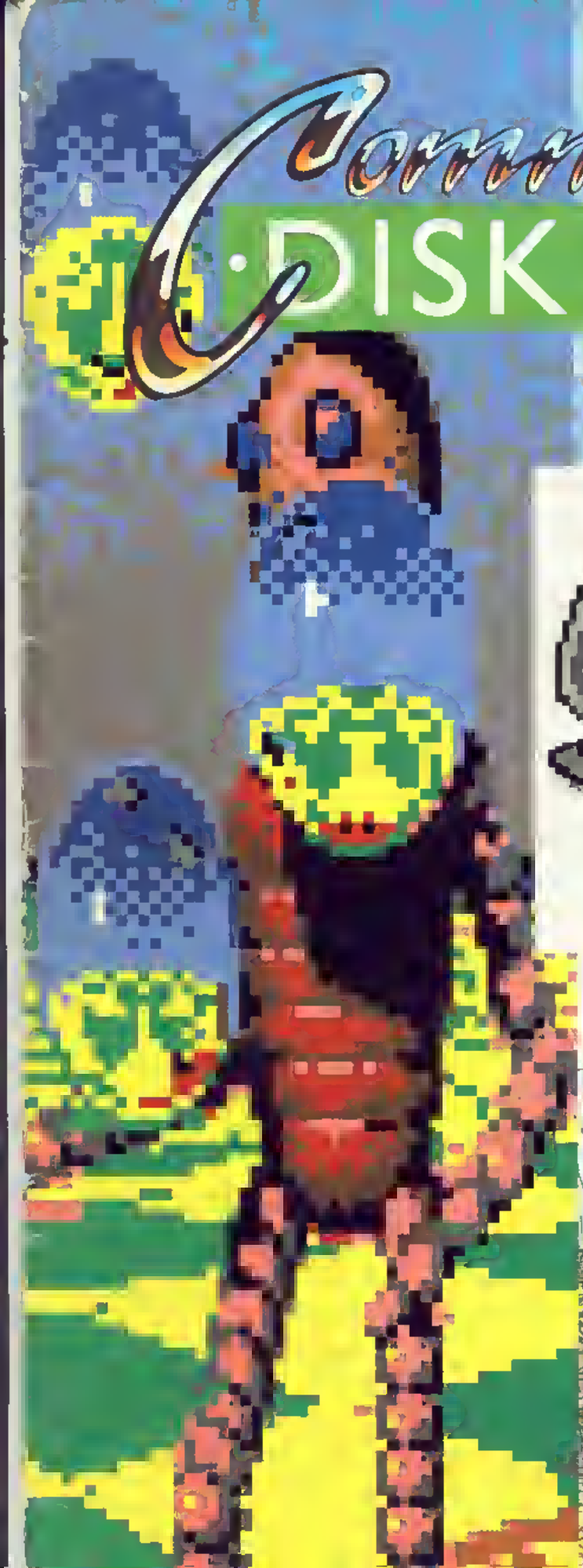
APRIL 1991 £3.25

Commodore DISK·USER



C64 ART POWER

THE DISK

Rescue

Screens V2

Word Count

Multi-Plexor

Hi-Res Converter

Screen Compiler

Colour Table Edit

6510+ Assembler

Char Ed+ Deluxe

04
9 770953 061014

LOGIC

London 081-882 4942 Cheshunt 0992 25323 Peterborough 0733 49696 London W1 071 935 2547

FULL RANGE OF AMIGA, ST, SEGA, 64, SPEC, AMSTRAD, PC, PCW, 2600, XL/XE, BBC

CAULDRON II	FIREBIRD	4.99
MUNSTERS	ALTERNATIVE	4 99
NAM	MINDSCAPE	7.99
SLDT CAR RACER	MINDSCAPE	4.99
BEYOND THE DARK CASTLE	MINDSCAPE	4 99
INDOOR SOCCER	MINDSCAPE	4.99
FINAL ASSAULT	EPYX	4 99
DIVE BOMBER	EPYX	6 99
SPY V SPY III	EPYX	4.99
SPORTS A RONI	EPYX	4 99
SPORTS NEWS BASEBALL	EPYX	5.99
DEATH SWORD	EPYX	7.99
IMPOSSIBLE MISSION	EPYX	2 99
THE GAMES		
SUMMER EDITIDN	EPYX	6 99
4*4 OFF ROAD RACING	EPYX	4.99
STREET SPORTS FOOTBA	EPYX	4 99
CALIFORNIA GAMES	EPYX	4.99
WDRLD GAMES	EPYX	4.99

STREET SPORTS BASEBALL	EPYX	4 99
SPACE STATION OBLIVION	EPYX	7.99
AXE OF RAGE	EPYX	4 99
SUMMER CHALLENGE	MINDSCAPE	4.99
WINTER CHALLENGE	MINDSCAPE	4 99
TRANSFDRMERS	MEDIAGENIC	4.99
CHAMPIDNSHIP BASEBALL	MEDIAGENIC	4.99
LEATHER GODDESSES OF PHOBOS	MEDIAGENIC	4 99
EYE OF HORUS	LOGDTRDN	3.99
STARRAY	LOGOTRON	3.99
FOOTBALL MANAGER	PRISM	4.99
ADDICTABALL	ALLIGATA	3.99
FOOTBALL MANAGER 2 CDNSTRUCTION KIT	PRISM	8.95
NDRTH STAR	GREMLIN	2.99

INTERNATIONAL SDCCER	CBM	4 99
VENOM STRIKES BACK	GREMLIN	2 99
GARY LINEKARS SUPERSKILLS		3.99
STARCROSS	CBM	4.99
SUSPENDED	CBM	4 99
DEADLINE	CBM	4.99
PARADROID	HEWSON	2 99
MURDER OFF MIAMI	CRL	4 99
UP PERISCOPE	SUB LOGIC	9.95
RUNNING MAN	GRANDSLAM	4.99
SUPER SCRAMBLE SIMULATDR		4 99
GLIDER PILOT	CRL	4 99
HERCULES SLAYER OF THE DAMNED	GREMLIN	2.99
BAAL	PSYGNOSIS	4 99
EXOLON	HEWSON	2.99
SHIRLY MULDOWNEYS TOP FUEL CHALLENGE	US GOLD	4.99
BLASTERDIDS	IMAGE WORKS	3 99

SDUND STUDIO HDME RECORDING STUDID	4.95
GEDS INC GEO WRITE GRAPHICS OPERATING SYSTEM WITH WORD PROCESSOR	9 95
SOUND EXPANDER FM SOUND MODULE	9.95
ASSEMBLER DEVELOPER	9 95
PROGRAMMERS UTILITIES	9.95
LOGO	9 95
PROGRAMMERS TOOL BOX	4.95
INTRO TO BASIC PART 2	4.95
PACK DF ACES	4 99
INTERNATIONAL KARATE, BDULDERDASH WHO DARES WINS, NEXUS	
BEST OF ELITE	4.99
BOMB JACK, FRANK BRUNOS BOXING SPACE HARRIER, AIRWDLF	
POWER HDUSE	2.99
FIGHT NIGHT, OSMIUM GOLDEN OLDIES	6.99
BIATHLON, MOONSWEEPER, LAWN TENNIS SPACE GALLERY, SLALOM, INTRUDER, SQUASH	
KO BOXING ETC ETC ETC ETC KICK BUT SLAM	4.99
FIST, RAMBO, UCHI MATA, BOP N WRESTLE PERSONAL MONEY MANAGER	9.95

BALLISTIX	PSYGNDSIS	4.99
NINJA HAMPSTER	CRL	3.99
ARTURA	GREMLIN	4.99
APACHE STRIKE	ACTIVISION	4 99
PAZZAZ	CBM	4 99
SCRAMBLE SPIRITS	GRANDSLAM	5.99
DEJA VU	MIRRORSOFT	8 99
SOKO BAN	SPECTRUM HOL	4.99
MURDER BY THE DOZEN	CBS	7 99
PARRALAX	OCEAN	4 99
INFILTRATOR II	MINDSCAPE	5.99
THE FLINTSTONES	GRANDSLAM	4.99
MS PACMAN		4.99
THUNDER CHDPPER	SUB LOGIC	9.95
UNINVITED	MINDSCAPE	7.99
PACLAND	GRANDSLAM	7.99
TERRYS BIG ADVENTURE	GRANDSLAM	5 99
BLOODWYCH	IMAGE WORKS	6.99
FIGHTING SOCCER	ACTIVISION	4.99
NAVY SEAL	COSMI	7.99
INTRIQUE	MIRRDRSOFT	6.99
BATALLIDN COMMANDER	SSI	6.99
THE THREE STOOGES	MIRRORSOFT	5.99

SWIFT SPREADSHEET 9.95

THE MOST POWERFULL SPREADSHEET PROGRAM
AVAILABLE ON THE COMMODORE 64, WITH POP
UP MENU CONTRDL, CELL MATRIX FRDM A1 TO
Z254 OR LARGER ON THE C128.

SUB BATTLE SIMULATOR 9.95

THIS IS YOUR CHANCE TO EMBARK ON WHAT IS
UNDUESTIONABLY THE MDST DETAILED REALISTIC, ALL ENCOMPASSING WAR SIMULATIDN EVER CREATED.

TRACK AND FIELD 9.95

THE CLASSIC SPORTS ARCADE GAME BY ATARI
SOFT INCLUDING A FREE ARCADE STYLE CDNTRDL PAD.

JET 11.95

BY SUBLOGIC FLY AN F16 FIGHTING FALCDN
THE MDST ADVANCED TACTICAL FIGHTER IN
THE WORLD, OR PUT YOURSELF AT THE CONTROLS OF A CARRIER BASED F1 8 HORNET
THE US NAVYS NEWEST MULTI ROLE FIGHTER

STEALTH MISSION 14.95

STEALTH MISSION PUTS YOU IN THE PILDTS SEAT OF THREE DIFFERENT JETS FROM THEF19, X29, OR NAVY TOMCAT.
NEW 3D ANIMATION TECHNIOUES PROVIDE DRAMATICALLYFASTER FRAME RATES FOR ALL COCKPIT WINDOW VIEWS
INCLUDES COMPLETE COR, ILS ADF AND DME AVIONICS FOR CROSS CDUNTRY NAVIGATION
COMPATIBLE WITH SUBLOGIC SCENARY DISCS.

WHERE TO BUY:
EITHER AT

19 THE BROADWAY
THE BOURNE
SOUTHGATE
LONDON
N14 6PH

UNIT 6
MIDGATE
PETERBOROUGH
CAMBS
PE1 1TH

5 LYTTON
PARADE
CHESHUNT
HERTS
EN8 8LF

MAIL ORDER TO:
5 LYTTON PARADE
CHESHUNT
HERTS
EN8 8LF
Tel: 0992 25323

POSTAGE AND PACKING: 1-3 ITEMS 75P, 4 OR MORE £1.00

CONTENTS

Commodore DISK·USER·

Volume 4 Number 6 APRIL 1991

ON THE DISK

CHAR ED DELUXE
A professional character designer 6

RESCUE
One for shoot 'em up addicts 10

SCREENS V2
An updated version of an earlier program 11

MULTIPLEXOR
Get those sprites zapping around the screen 16

HIRES CONVERTER
Convert your characters with ease 17

SCREEN DESIGNER/COMPILER V2
Yet another update to an earlier program 19

COLOUR TABLE EDITOR
The correct version of this superb utility 35

6510+ ASSEMBLER
Reproduced to compliment the new series 36

IN THE MAGAZINE

WELCOME
Instructions and Editors comment 4

ADVENTURE HELPLINE
More help with THE ASTRODUS AFFAIR 9

BASICS OF BASIC
Basic programmers get more assistance 14

C64 ART POWER
Art packages for the C64 analysed 22

EASY WRITER REVIEW
A 3 in 1 office package is put to the test 24

TECHNO INFO
Jason Finch answers some more mind testers 28

INTRODUCTION TO MACHINE CODE
A new series begins for budding M/C programmers 42

Publisher: Hasnain Walji
Group Editor: Paul Eves
Technical Editor: Jason Finch
Publishing Consultant: Paul Crowder
Advertisement Manager: Cass Gilroy
Classified Sales: Deborah Curran
Designer: Mark Newton
Distribution: Seymour Press Distribution
Ltd. Windsor House, 1270 London Road, Norbury, London SW16 4DH. Tel:081 679 1899. Fax: 081 679 8907
Printed By: Gibbons Barford Print

Subscription Rates

UK £[illegible].00
Europe £39.00
Middle East £39.30
Far East £41.60
Rest of World £39.70 or $69.00
Airmail rates on request
Contact: Select Subscriptions. Tel: (0442) 87666

Commodore Disk User is a monthly magazine published on the 3rd Friday of every month. Alphavite Publications Limited, 20, Potters Lane, Kiln Farm, Milton Keynes, MK11 3HF. Telephone (0908) 569819 FAX. (0908) 260229. For advertising ring (0908) 569819

Opinions expressed in reviews are the opinions of the reviewers and not necessarily those of the magazine. While every effort is made to thoroughly check programs published we cannot be held responsible for any errors that do occur.

The contents of this publication including all articles, designs, drawings and programs and all copyright and other intellectual property rights therein belong to Alphavite Publications Limited. All rights conferred by the law of copyright and other intellectual property rights and by virtue of international copyright conventions are specifically reserved to Alphavite Publications Limited and any reproduction requires the prior written consent of the company.

© 1991 ISSN 0953-0614

EDITORS COMMENT

IN THE WORDS OF THAT GREAT ENTERTAINER, DAVID FROST, I SAY TO YOU ALL - HELLO, GOOD MORNING AND WELCOME. THIS MONTHS ISSUE OF YOUR FAVOURITE C64 MAGAZINE IS FULL OF GOODIES TO HOPEFULLY PLEASE EVERYONE.

FOR THE NOVICE MACHINE CODE PROGRAMMER, WE START AN IN DEPTH SERIES ON GETTING TO GRIPS WITH THIS FASCINATING SUBJECT. THE IDEA IS THAT IT WILL NOT BE AS COMPLICATED AS MOST OF THE TEXT BOOKS ON THE SUBJECT. TO COMPLIMENT THIS, WE HAVE REPRODUCED THE EXCELLENT ASSEMBLER PROGRAM. 6510+ WHICH WAS PUBLISHED A WHILE BACK

THERE ARE THREE PROGRAMS ON THE DISK WHICH ARE UPDATES OF PREVIOUS ONES. (THE IDEA BEING THAT AN UPDATE IS THAT MUCH BETTER THAN THE ORIGINAL).

BY FAR THE MOST EXCITING PROGRAM THOUGH IS THE CHARED DELUXE UTILITY THIS IS A VERY PROFESSIONALLY PUT TOGETHER CHARACTER EDITOR WHICH MANY OF YOU WILL FIND INVALUABLE IN YOUR LIBRARY OF UTILITY PROGRAMS

THE FRONT COVER DEPICTS SOME OF THE KIND OF THINGS YOU CAN DO WITH VARIOUS ART PACKAGES. THESE ARE BROUGHT TO YOU COURTESY OF JENNI SIMPSON (THE WRITER OF THE ARTICLE - C64 ART POWER). ENJOY THIS MONTHS MAG.

DISK INSTRUCTIONS

Although we do everything possible to ensure that CDU is compatible with all C64 and C128 computers, one point we must make clear is this. The use of 'Fast Loaders', 'Cartridges' or alternative operating systems such as 'Dolphin DOS', may not guarantee that your disk will function properly. If you experience problems and you have one of the above, then we suggest you disable them and use the computer under normal, standard conditions. Getting the programs up and running should not present you with any difficulties, simply put your disk in the drive and enter the command

LOAD" MENU",8,1

Once the disk menu has loaded you will be able to start any of the programs simply be selecting the desired one from the list. It is possible for some programs to alter the computers memory so that you will not be able to LOAD programs from the menu correctly until you reset the machine We therefore suggest that you turn your computer off and then on again, before loading each program.

HOW TO COPY CDU FILES

You are welcome to make as many of your own copies of CDU programs as you want, as long as you do not pass them on to other people, or worse, sell them for profit. For people who want to make legitimate copies, we have provided a very simple machine code file copier. To use it, simply select the item FILE COPIER from the main menu. Instructions are presented on screen.

DISK FAILURE

If for any reason the disk with your copy of CDU will not work on your system then please carefully re-read the operating instructions in the magazine. If you still experience problems then:

Within eight weeks of publication date disks are replaced free.

1. If you are a subscriber, return it to:
Select Subscriptions Ltd
5, River Park Estate
Berkhamsted
Herts
HP4 1HL
Telephone; 0442 876661

2. If you bought it from a newsagents, then return it to:
CDU Replacements
Stanley Precision Data Systems Ltd
Unit F Cavendish Courtyard Sallow Road
Weldon North Industrial Estate
Corby
Northants
NN17 1JX
Telephone; 0536 61787

After eight weeks a replacement disk can be supplied from STANLEY PRECISION DATA SYSTEMS for a service charge of £1.00. Return the faulty disk with a cheque or postal order made out to STANLEY PRECISION DATA SYSTEMS and clearly state the issue of CDU that you require No documentation will be supplied. Please use appropriate packaging, cardboard stiffener at least, when returning disk. Do not send back your magazine, only the disk please.

NOTE: [illegible] disks only to the above address if [illegible] that does not appear to work only if the disk is faulty. Program faults should be sent to BUG FINDERS, CDU, Alphavite Publications Ltd, Unit 20, Potters Lane, Kiln Farm, Milton Keynes, MK11 3HF. Thank you.

OFFICIAL DEALER FOR ☆ COMMODORE ☆ STAR ☆

COMPUTERS

Amiga 1500 comprising : B2000+1084S Monitor+Twin Floppies+ The Works!+Deluxe Paint 3+3 Games	£928.00
Amiga 500 SCREEN GEMS PACK comprising : TV Modulator, 4 Games & Deluxe Paint 2	£359.00
Amiga 500 Basic Pack comprising : TV Modulator, Mouse, Workbench 1.3, Extras Disk, Tutorial, Manuals	£319.00
Commodore 64C NIGHT MOVES Pack comprising : Cassette Recorder, 8 Games & 2 Joysticks	£149.95
Commodore PC Starter Packs(AS SEEN ON TV)	PHONE.

PRINTERS

Citizen 120D+Perallel or Commodore	£129.95
Star LC-10 Parallel	£159.00
Star LC-200 Colour Parallel	£209.00
Star LC 24-10 24 pin Multi-font170/57cps	£209.00
Star LC 24-200 24 pin Multi-font 200/67 cps	£259.00
Star LC 24-200 Colour Version of above	£299.00
Okimate 20 Print heads/Ribbons/Paper	PHONE.

MONITORS

Commodore 1084S Stereo Colour Mon	£249.00
Philips 8833-II Stereo Colour Monitor	£249.00
Philips 7502 Green Screen Monitor	£ 95.00

MISCELLANEOUS

Amiga 512K RAM/Clock Exp for A500.Free p&p.		£ 39.00
Commodore 1541-II Disk Drive		£129.00
Commodore C2N Data Recorder.	p&p. £2	£ 29.95
External 3.5" Disk Drive for Amiga	p&p. £2	£ 59.95
Power Supply for C64	p&p. £1	£ 26.45
Super-G Parallel Interface for C64/128	p&p. £1	£ 34.95
(Allows use of stenderd parallel printers on C 64/128)		
User Port Parallel Printer Cable	p&p, £1	£ 15.99

SOFTWARE

Superbase 64 or 128	Free p&p.	£ 39.95
Superscript 64 or 128	Free p&p.	£ 39.95
Tasword 64 40/80 Col WP-Tape or Disk	Free p&p.	£ 24.95
Mini Office II for 64 - Disk	Free p&p.	£ 19.95

ALL PRICES ARE INCLUSIVE OF VAT AT 15% CARRIAGE £5 (EXPRESS £10)
Prices subject to change without notice

Delta Pi Software Ltd

VISA

8 Ruswarp Lane, WHITBY, N. Yorks YO21 1ND
TEL/FAX: 0947 600065 (9am - 7pm)

MasterCard

READERS' PAGE

From next month we will be offering the facility of a Readers' Page, where for a small handling fee, you can advertise all those old or unwanted items that you are waiting to get rid of, and make some spare cash.

This facility is not open to trade advertisers, only to private individuals.

The maximum number of words per entry is twenty. Your entry will be printed in the next available issue.

HOW TO PLACE YOUR ENTRY

All you have to do is send off the form on the right, together with your payment; Entries cost just £5.00 each.

ENTRY FORM

POST TO: **Debbie Curran, CDU Readers' Page, Alphavite Publications, 20 Potters Lane, Kiln Farm, Milton Keynes MK11 3HF**

Please place my ad in the next available issue of CDU.

Name.....

Address .....

.....

.....

Method of payment: ❑ Cheque ❑ P.O.

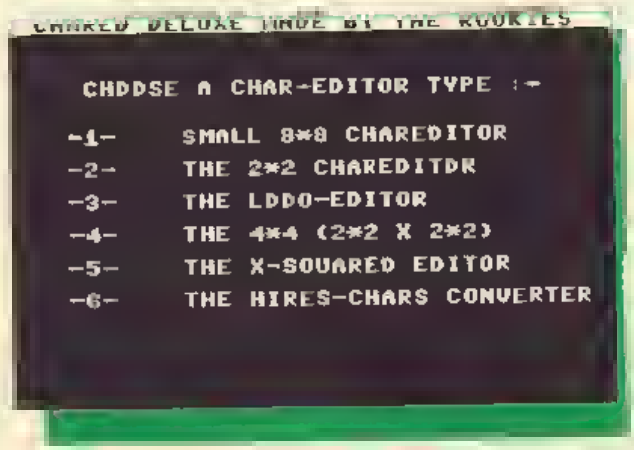
Please make cheques and postal orders payable to **ALPHAVITE PUBLICATIONS LTD.**

Write your Advertisement here, one word per box. Include your name, address and phone number if you want them printed.

CHAR-ED DELUXE

A professional looking character editor gets an airing

FRANK JOPPE



=Character Editors and Sprite Editors abound in their thousands. Most are adequate but lack one or two refinements. I have decided to let you, the readers of CDU, share in the joy of using this Character Editor that I have devised.

LOADING INSTRUCTIONS

Switch on the machine, type LOAD "CHARED DELUXE ",8,1 or select it from the CDU menu and press RETURN. The program starts automatically.
IMPORTANT: You must NOT have a cartridge inserted !!! this WILL disturb some of the editors !!!
When you get past the intro you will see a menu where you can choose which editor you like to use. Pressing 'H' in the CHAREDIT modes for all programs will show the help screens. Pressing 'H' in BLOCKed mode won't work. You must get the key commands from the help-screens as I'm going to explain some memory management for some editors here.

THE 8*8 EDITOR

This editor allows the user to make CHARS in an 8*8 (or 4*8 in multi colour) grid. The rest is simply straight forward. It is very much like FONT FACTORY, but it saves only one charset. (FONT FACTORY saves 2 charsets).

THE 2X2 EDITOR

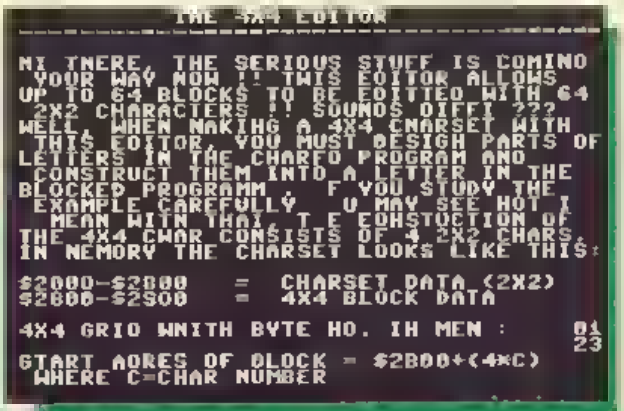
This editor allows you to draw chars in a 16*16 grid, or 2x2 chars. When putting the chars on the screen you have only 64 characters available I'll explain in binary;
76543210 = one BYT. bit 6 represents 64 (and bit 7 128) so, bits 0-5 are used for the character number. Bits 6 and 7 represent the level of the character and have a maximum of 3. So 4 characters can be selected by bits 6 and 7, you can do this simply by oring $40 or $80. In a character, bit 6 is on at the right most half of the chat and bit 7 is on at the bottom half of the char.
Look at FIGURE 1, it describes the value of bits 6 and 7 in a char. In the example you can see, at the left - the memory addresses (VIDEO RAM), at the right and at the bottom - the OR codes, the upper left character is of course the main character, between $00-$3f. So if you write a scroll in ASCII you must AND the letter with $3f (=63).

THE LOGO EDITOR

This editor is like ULTIMATE FONT EDITOr, look at the example. Look also at FIG. 3. You can see here how the characters are arranged on the screen. The logo you draw is 32 chars wide and 8 chars high, but then you have (40-32)/2=4 characters at both sides of the logo (because screen width is 40 chars !!!) Well, to cover the sides of the logo, and the row above and the row below the logo, you must give these locations the same colour as the background, in COLOR RAM of course. This is the reason why you can use only the first seven colours. More info about this under the converter

THE 4*4 (2X2*2X2) EDITOR

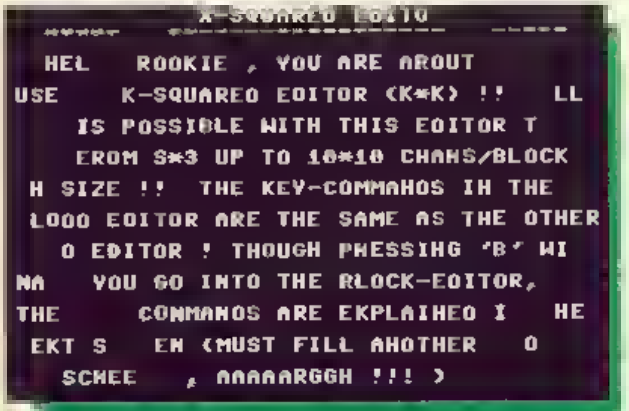
This is a combination of two editors, the 2x2 editor and a block editor.
When drawing with this program, you draw parts of bigger characters in the 2x2 chared program and construct them together in the block editor.
$2000-$27FF is used for the charset and $2800-$28FF is used for the 4x4 blocks. Formula, SA=$2800+(CH*4) where SA = start address of block and CH = block number. The positions on the screen of the block bytes are similar to the values of bits 6 and 7 in FIG. 1.The value must be added up by SA. See FIG 2 for more info about this. The block editor is used in a quite simple way, you select a char from your 2*2 set, position the flashing cursor, and you press space. The particle is placed under the cursor. Current Char tells you which BLOCK you are editing, NOT the particle



THE X-SQUARED EDITOR (X*X)

This is quite a complex one!! It is like the 4*4 but you work with small chars and you work with a selectable size. Smallest chars 3*3 and biggest chars 10*10, everything in between is also possible. You must read the help screens carefully this time for the BLOCKed .The BLOCKed works like the one in the 4*4, but you can go UP/DOWN/LEFT/RIGHT in the charset with the function keys. In memory, depending on the size of the blocks, the blocks are stored in y * x bytes of memory where y is of course the height and x the width. So the first x bytes are the top row/column (I don't know the difference !!). The second x bytes are for the layer under the top row/column (aaargh !). The memory is arranged $2000-$27FF = charset

$2800-$2800+(y*x)*64 block data !!
NOTE; when changing the size of the block you must re-make the char, it won't expand automatically !!
The bigger the size, the more memory used, the biggest set 10*10 uses 6.25K RAM for the block data only!! Plus the charset = 6.25+2=8.25K bytes!! that is 8.25*4 (1K byte = 4 blocks) 33 blocks!! On disk it's about 36 blocks



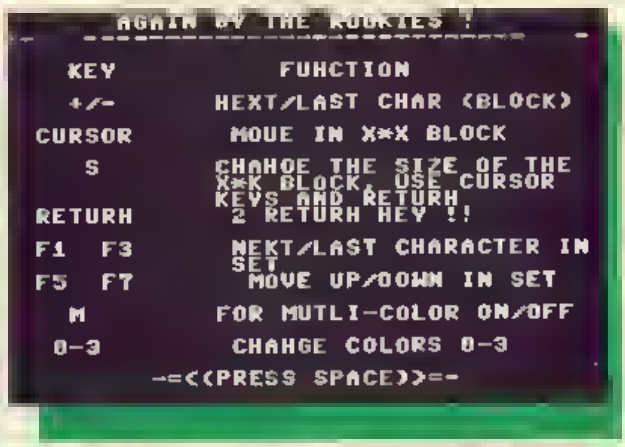
THE CONVERTER

This very handy program allows you to convert your hires pictures into a charset + screen. The limit is that KOALA painter pictures only can be converted, as this is the standard for all hires picture files. The converter comes in handy for graphic text adventures or, you can see a very good example, for demo's and intro's !! Load in the picture and select the numbers for each colour.
0 = colour stored in background (in multi-lo-res mode) $D021 = 53281
1 = colour stored in $D022 = 53282
2 = colour stored in $D023 = 53283
3 = colour stored in $D800 - $DBFF

NOTE; in lo-res it is possible to use multi-colour characters, but, the characters can only use colours 0-7 in colour-ram, if you AND the colour with 7, the character will be standard, if you AND with 7, OR with 8, the character will be in multi-colour, try it, this is really true !! It is also the reason why you shouldn't put a '3' above one of the bottom colours (in the convert mode) because they can't be used, unless of course you plan to give them another colour.

TO CONCLUDE

I made these cool character editors because ii was annoyed about the fact that I had some char editors but they were spread on various disks, ii I'd have copied all of them onto one disk then I wouldn't be pleased, yet because programs like the 4*4 editor and the X-squared editor weren't in my library. At last, I can ask our painter to make a BIG charset, which he couldn't make before..."How do they do it ?" he asked me often enough, well, with this utility!
You might wonder why I put the other editors in as well, well, I said I was annoyed about the fact etc. I wanted to make a COMPLETE utility, now you can scratch your other 3 million char editors because this utility contains everything you need...Oh , yeaaah, if you use one of the utilities to make a charset or so for a demo or a game, don't forget to greet me hey!



ADVENTURE HELPLINE

That devious adventure 'ASTRODUS AFFAIR' gets another grilling from JASON FINCH

"The familiar darkness encloses as death overcomes you". Well, that is the message that I hope none of you will be seeing in THE ASTRODUS AFFAIR anymore. We have again reached the point where I let you know a few of the secrets of this excellent adventure. Last month we embarked on the first actual problem-solving session, which we shall continue in the same style this month. We will cover three of the problems that you will face and will now be able to overcome. One of those is the task of repairing the hole in location 15, which I believe is one of the most complex puzzles, relying on a number of events to have occurred before you can mend it. First of all we'll take a look at the mouth in the door!

THE HUNGRY DOOR!

South of the generator room you will have come across a closed door that had a full set of teeth installed, ready and waiting for something! But what could these teeth be waiting for? Food, of course! So, from where we left off last month, you will have to find the food on the middle level and then go upstairs, passed the sleeping TORTOR, and to the place where this door/mouth is situated. On the way, nip into the generator room again and collect the wire mesh that is there - it will prove useful later on. When you are ready, just give the food to the mouth and it will miraculously transform itself into a little creature called a SLOFT, which then flees from the room. You are then able to go through the door. But STOP - you don't want to just yet because there is no point. Instead, go back to the Bridge, then go south and you will find the SLOFT, cowering in a corner. Examine the manual that you found in the drawer last time, and using the information in it, switch on the heating, then find a staircase going to the lower level and go down it. Here lies another problem. But first of all, for anyone who didn't get all that about the mouth, there follows the systematic breakdown of moves which I will try to get in the right order. If you want to try it yourself, please skip the next section.

FEEDING THE MOUTH

OPEN the DRAWER and TAKE the MANUAL that you discovered last time. Now go EAST twice, then NORTH, then WEST, and then NORTH again. TAKE the FOOD and return SOUTH to the crew's area. Now go EAST, SOUTH and EAST again. Now you should be at the bottom of the staircase. Go UP it and then NORTH when you reach the top. TAKE the WIRE and return SOUTH. Go SOUTH again and GIVE the FOOD to the mouth. Now that that is done, move NORTH and go back DOWN the staircase. Go WEST, WEST again, and WEST a third time to get back to the Bridge. Now go SOUTH and EXAMINE the MANUAL. This will tell you to PRESS R42D to switch the heating on - do it. Go NORTH and the SLOFT will hopefully follow you. Now go EAST and EAST again, SOUTH and then DOWN a staircase

SLOFT AND DRYGARS

Without the SLOFT, if you were to proceed south to location 22 from here, you would soon be finished off by a group of hungry drygars. But with the SLOFT, you can go south safely. The drygars will spot the SLOFT and dive for it. It then heads into the supplies room and you follow. Somehow the SLOFT metamorphosises, grabs a laser from the ceiling and then shoots one of the drygars. The others flee in terror and you are safe. You can go and grab everything that is useful, even anything that is broken, and drop anything that you don't need. Somewhere nearby you'll find some sealant. When you have that, return to the room south of the Bridge. Ignore the next section if you don't want to know how to get the drygars away step by step.

'USING' THE SLOFT

Go SOUTH from the bottom of the staircase and let the SLOFT do its stuff! When in the supplies room, location 21, TAKE the FILINGS, TAKE the VISIONISER, and TAKE the TORCH. You can DROP the CARD and DROP the MANUAL - they are of no further use really. Now go EAST and then SOUTH to find the SEALANT. Return to the middle level by going NORTH and then UP the staircase. Then go NORTH, then WEST, where you will see the fate of the fleeing drygars, WEST again and then SOUTH.

MENDING THE HOLE

The key to mending the hole is the silver disc on the floor of the corridor where the hole is. The gravity globe, if in that location, will not move off its spot, and if you are holding onto the globe, then you will not be sucked

out of the hole. There is an added problem because it is difficult to move the globe off its original silver spot in one of the other locations. You must fill the visioniser with the steel filings to operate it, shut down the system and then get the globe. You can then power the system up again, by pressing the appropriate key on the control panel. It is then possible to electrocute the drygars with the broken torch (examining it will reveal its potential!), and you are then free to repair the hole. The globe which you are clinging to will not be sucked into space, and consequently neither will you. With the mesh and the sealant in place, all is well! Close your eyes if you want to do it on your own!

DIY REPAIRS

PUT the FILINGS IN the VISIONISER and then PRESS A51X. This shuts the system down and you need the visioniser to provide light. Then go NORTH, EAST, EAST and NORTH again TAKE the gravity GLOBE (which you can do because the power is off so it isn't attracted to its spot) and return to the control room by moving SOUTH, WEST, then WEST, and SOUTH again. PRESS A51X to restore full power. Now go NORTH and EAST - the globe will not allow you to go any further because in location 15 there is a gravity disc on the floor and power is now on. So simply ATTACK the DRYGARS and REPAIR the HOLE. It's simple when you know how!

GOOD LUCK

That, unfortunately, is all we have time for this month. Next month I'll be looking at a few more of the problems that you may come across in this truly first class adventure. Maybe one day you will able to fly the Astrodus - we can all live in hope at least!

RESCUE

Travel the galaxy and rescue the humanoids DAVID BRYSON

Far into the future, the Homosapien race has innovated and expanded in technology and occupation so much that they now are inhabiting and controlling several planets and moons with the aid of an immense communications network based on the planet Earth, (which has gained acclaim for it's reliability, but certainly not it's cost).

MEANWHILE

Meanwhile humanoids have stolen the jobs of many people because of the arrival of this extravagant technology that is cheaper, more efficient and more adaptable to harsh environments. The capitalist government's trust of the security of the network soon diminishes when computer thugs, In league with the ex-miners, manage to crack the security system and uphold all mining activity on the eight planets and moons. The government is not prepared to turn this into a scandal as it has put so much money and effort into the network So they ask you, one of the top ranking space pilots of the age, to travel around the moons and planets and rescue the malfunctioned humanoids, to be awarded a substantial sum of money and pay no taxes for life. You can't resist the opportunity, and head off in the direction of the colonies.

PLAYING RESCUE

Plug your joystick into PORT 2 and control the ship by pushing it left to thrust left and right to thrust right. Up controls the upwards thrust and down controls the downwards thrust. (Isn't life simple?) Pressing FIRE does not do anything at all. Precise timing and co-ordinated control of the joystick is required if you want to complete all 8 levels, but I must say they do not progress in difficulty therefore keep trying in the early stages. When you go near a humanoid, it jumps up and down ready for rescuing, which you do simply by running past it. If you want to go and meditate, press "P" to pause the game. If the mission gets too stressful, press "Q" to quit and go and have a lie down for a few hours.

TECHNICAL DETAILS

The game is protected against SPRITE-BACKGROUND disabling and infinite lives functions on cartridges, so just pull that nasty protrusion out of the back and chuck it away The game code is over 3K long and the graphics takes up over 4K. The screens take up 8K (and no, they are not compresses, who need to when the program gets compressed anyway). I hope you enjoy the game

SCREENS V2.1

Extra screen storage, Fast data writer and Disk turbo loader from one program

PHILLIPE BASTINGS

If you bought the MAY 1990 issue of CDU, then you will now be developing your Basic programs using the utility called SCREENS which was included on the disk. This original program was lacking in some much needed functions, therefore I now present you with the latest version which includes 16 functions, instead of the original 8 plus a FAST DATA WRITER and DISK TURBO LOADER. Let us take a closer look at all the functions and see how easily they are accessed. All the power of this package is at the top of your fingers. The function keys are redefined in this way;

	KEY	+ SHIFT	+ C=	+ CTRL
F1	PREV LINE	CLEAR TOP	INSERT LINE	DELETE CHAR
F3	NEXT LINE	CLEAR BOTTOM	DELETE LINE	LIST
F5	PREV PAGE	CLEAR SCREEN	CLEAR LINE	LIST + CR
F7	NEXT PAGE	CLEAR ALL	CLR LN KEEP N	RUN:

If you did not buy the above mentioned issue of CDU, then you do not know that SCREENS adds an extra 2Kb of SCREEN storage to the usual screen. The functions F1, F3, F5 and F7 allow you to restore screen lines or pages that have disappeared because of the scrolling of the screen. What you did not have with SCREENS V1 are functions to manipulate the REAL screen! This is why SCREENS V2.1 was born.

FUNCTIONS AVAILABLE

F2 - CLEAR TOP Will clear your screen from the top to the actual cursor position and place the cursor home allowing you to enter new program lines while other lines are displayed at the bottom of the screen.

F4 - CLEAR BOTTOM is the opposite of the previous described function. It will clear the screen from the actual cursor position to the end of the screen.

F6 - CLEAR SCREEN is the same as pressing the keys <SHIFT><CLR/HOME>.

F8 - CLEAR ALL will clear the actual screen as well as the 2 extra screens! Note that this function is automatically performed at initialisation.

F9 - INSERT LINE (C= + F1) allows you to move down a part of the Basic text and insert new lines.

F10 - DELETE LINE (C= + F3) will delete the line were the cursor is. This function is the opposite of the previous one and performs a scrolling up of the screen starting at line 24 and ending at the actual cursor line.

F11 - CLEAR LINE (C= + F5) will clear the actual line and place the cursor at the start of the line.

F12 - CLEAR LINE KEEP NUMBER (C= + f7) is a function you won't find in any text editor or even in any wordprocessor! The reason is very simple, yours truly has invented it just for you! What the function does is very simple. It checks whether or not the current line begins with a Basic line number. If not then nothing happens, but if it does the line number is kept, the rest of the line is deleted and the cursor is placed just after the last number allowing you to type in new statements! This function is very useful if you start typing statements and suddenly realise you are completely wrong. This is also a precious function if you want to delete several lines of your Basic program. You only need to place the cursor at the start of the line you want to delete, use function F13 and press RETURN. If your Basic line is not longer than 40 characters it will be deleted.

FUNCTION DEL CHAR (CTRL + F1) is a replacement for the actual DEL key of your keyboard which is in fact a mixture of DEL CHAR key and a backspace key! This function, F13, will move the text one character to the left but the cursor position remains unchanged.

Functions F14, F15 and F16 do not need any explanation. Their names tell you exactly what they do.

FAST DATA WRITER

As a C64 programmer, I also type in programs found in magazines. Most of these are machine code programs using a Basic loader. When you enter a Basic loader, you always type the line number, the DATA statement, commas and figures. What is time consuming is to go to the COMMA key that is located on the LOWER row of the keyboard while the numerical keypad is on the UPPER row! The second problem is that the numerical keypad is disposed horizontally.
My idea was to replace the left arrow key, just near the "1" key, and the "+" key, just near the "0" key, by a COMMA. So when you have entered a value, use the closest COMMA key. The "." key has been replaced by the DATA statement. Instead of typing four characters, you only need to hit one key. All this is done while your hand moves from left to right and not left,

right, down, up . . I can guarantee that after using FDW for five minutes, you will never again enter a Basic loader without using it!
The following question has perhaps raised in your mind. "What will happen if I really need the left arrow key or the + and - keys? To solve this problem, FDW works with a switch. Hit the following keys simultaneously <CTRL><C=><F1>. An audible beep will confirm to you that FDW has been switched ON/OFF. When FDW is ON you still can have the normal key by keeping your finger a little longer than usual on the key. Using the DEL CHAR function will delete the garbage that has been displayed. As a summary, when FDW is ON the upper row of the keyboard looks like this:

,1234567890,DATA

When FDW if OFF it reverts back to normal.

DISK TURBO LOADER

As previously mentioned, SCREENS V2.1 offers you a DISK TURBO LOADER which will load your programs about 6 times faster than normal. There is only one restriction, when loading a program, your printer must be switched OFF but you may switch it back ON again as soon as the program is loaded
You may also use SCREENS V2.1 to load your games as this package loads quite fast like a normal Basic program and installs itself into memory instantaneously. When the TURBO LOADER is active, your screen will be flashing whilst loading programs into memory.

MEMORY USAGE

SCREENS V2.1 uses the area from 49152-53130 ($C000-$CF8A) for the functions and the disk turbo loader. It also uses 3000 bytes of memory beneath the Basic interpreter to store the 2 extra screens and a 1Kb buffer. This should not disturb Basic programs as long as they do not exceed 35Kb which is quite a lot for a program, Note that when a Basic program starts running, SCREENS V2.1 will disable itself and re-enable automatically when your Basic program stops. If you hit <RUN/STOP><RESTORE>, SCREENS V2.1 will be deactivated. You can always restart the program by typing SYS49152. This utility loads just like a normal Basic program but installs itself into memory using a machine code loader.

WORD COUNT

Keep tabs on your epic essays SIMON COLLIS

Certain word-processing programs include no way of actually counting the number of words you have written. Sometimes, this can be a blessing (in cases of writer's block, for instance, you can pretend you've written thousands of words, when in reality you haven't even hit double figures!).
I was half-way through a second chapter of a book when I pondered this difficulty. Not wanting to go back to the beginning of the document, and count all the words by hand (a strain on the eyes, but good for spotting typing errors!) instead, I got out my assembler, and wrote a small program in machine code to do just that task for me. The end product is WORD COUNT.

GETTING STARTED

To load WORD COUNT, either use the CDU menu, or type; LOAD "WORD COUNT",8; followed by RUN when it has loaded.
When RUN, WORD COUNT will present you with a menu of options. To use a command, type the command letter (shown in the left-hand column), followed by any parameters

THE COMMANDS

C - [FILENAME] This will open a file on the disk, and count the number of words in it. For the uses of this program, a word is defined as being a RETURN, or SPACE, following several non-space and non-RETURN characters.
Note that the default file-type is defined as SEQ. If your word-processor does not save the file as SEQ, then simply put ",P" or ",U" after the filename to use a different file-type and WORD COUNT will humbly accede to your request.

S - This command will show you the word-count status (seen automatically after a C command), which shows the word-count of the last file, and the total word-count of all files counted so far.

Z - The command, which needs no parameters, zeros the word-counts, both the last file count, and the total word-count so far.

@ - This allows you to use a DOS command, for example, VALIDATE, or INITIALIZE. All CBM-DOS commands are available, including $.

? - This asks for help, and prints a list of the 6 available commands.

Q - When you've word-counted all you want to word-count, this will take you back to BASIC by resetting the machine.

■ UTILITIES UNLIMITED ■ UTILITIES UNLIMITED ■ UTILITIES UNLIMITED ■

UTILITIES UNLIMITED

THE WORLD'S BIGGEST PROVIDER OF 64/128 UTILITIES

■ UTILITIES UNLIMITED ■ UTILITIES UNLIMITED ■

LOCK PICK BOOKS

See the internal workings of a parameter. Each book contains step-by-step instructions. These books use HesMen cartridge

LOCK PICK BOOK 1

Covers 100 popular software titles, including GEOS and Newsroom. Lock Pick Book 1 includes FREE Utilities Unlimited Krack & Smash, etc., etc. Lock Pick Book 1 **£9.95**

LOCK PICK BOOK 2

A more comprehensive book explaining how and why parameters work and how different types of protection are used. Lock Pick Book 2 **£12.95**

SPECIAL OFFER

Lock Pick Books 1 & 2 PLUS HesMon Cartridge 64
£24.95

HESMON 64

A Machine Language Monitor cartridge for the 64/64C/SX or 64/128/128D. HesMon 64 is a special cartridge, it simply forces the computer into Machine Monitor/Enter mode. It is perfect for all monitor applications, including the Lock Pick Books. HesMon 64 only **£6.95**

THE PARAMETER CONSTRUCTION SET

New from Utilities Unlimited, the Parameter Construction Set 64 allows you to produce parameters for almost any piece of disk software. Using this easy to use, menu driven program you can read, compare and write parameters. For beginners a parameter is simply a routine which bypasses the software's built-in copy protection.

Detailed instructions guide you through constructing a parameter to copy a well known game. Includes instruction manual, parameter construction software **PLUS 100 free parameters**. Why wait for companies to release parameters when you can write your own. Parameter Construction Set **£19.95**

THE ORIGINAL PARAMETER CROSS REFERENCE GUIDES

These guides list over 10,000 programs and tell you how to copy every one. Details are given for upgraded programs. Different copiers are listed. Copiers covered include Maverick, Renegade, Lock Pick Books and many more.

QUARTERLY UPDATES

Keep up with continuing developments of both back-up producers and software developers. Book updated one quarter, disk the next and so on.

THE ORIGINAL PARAMETER CROSS REFERENCE DISK

Load the disk, type in the program you require — Hey Presto! the details are displayed to screen or printer. It also allows you to create a custom report of just the products you choose and print it out on your printer.

THE ORIGINAL PARAMETER CROSS REFERENCE BOOK

A large A4 size reference book with all 10,000 titles listed in alphabetical order. Simply locate the details and produce the back up. 150 pages.

The Parameter Cross Reference Book **£24.95** Disk **£12.95**
The Parameter Cross Reference Book & Disk **Only £29.95**

1000 PARAMETERS PACK

Utilities Unlimited have consolidated all their parameters into one menu driven program for the 64. 1000 parameters ready at the touch of a key.

Many companies are covered including: Abacus, Cinemaware, Datasoft, Epyx, Microprose, SubLogic, Timeworks and many more.

Included **FREE** is Utilities Unlimited's last Nibbler so the 6 disk pack is ready to run for only **£19.95**

THE 128 SUPER CHIP

There is an empty socket inside your 128 or 128D waiting for the SuperChip with 32K worth of great built-in utilities available at the touch of a key. Full fitting instructions and software manual are included — and no soldering required.

SUPERCHIP A includes a File Copier, Nibbler, Track & Sector Editor, Screen Dump and even a 300/1200 Terminal program which is Hayes compatible. Best of all no memory is used up by any SuperChip.

SUPERCHIP B contains Super 81 Utilities, the award winning, top selling utility pack just for the 128 and 1581 disk drives. Some of the many utilities are: Disk & File Copy, CP/M & MS-DOS utilities including Format, Drive Monitor, Disk Editor, RAM Writer, File Unscratch, Unlock & Lock, AutoBoot Maker and much more.

SUPERCHIP C is a combination of SuperChip A and B. The built in switch allows you to switch between each with ease and SuperChip C saves you money!

SuperChip A or B **£19.95**
SuperChip C . only **£34.95**

BLITZ DE-COMPILER 64

Have you ever wondered how a program was written? Blitz De-Compiler will uncompile a program back into a BASIC 2.0 listing with each command on a single line. Many English and American commercial packages are compiled from BASIC into speed code using Blitz. Blitz De-Compiler 64 **£12.95**

GRAPHIC LABEL MAKER 64

This program will allow you to use graphics from PrintShop to create high quality, impressive labels. Graphic Label Maker is easy to use, enter the label text, import the graphic and print out to your dot matrix printer. Graphic Label Maker 64 **£6.95**

MASTER LOCK 64

Have you ever wanted to protect your work from hackers? Master Lock 64 has two sections, the first encrypts your programs and the second adds a parameter to your disk. Each and every Master Lock is different so no one will ever be able to crack the protection. Master Lock 64 **£12.95**

PHOTO COPY 64

Photo Copy 64 is a graphics integration program used to transport your favourite graphics from one program to another. It can convert PrintShop to Newsroom, Hi-Res to PrintShop or Newsroom. Photo Copy works with a variety of files. Doodle, Flexidraw, PrintShop, Screen Magic and Computer Eyes. Photo Copy includes full instructions. Photo Copy 64 **£6.95**

THE ARTFUL DODGER 64

The Artful Dodger is a unique utility. Simply insert a back up (copied with any fast data copier) of any Electronic Arts package, press a key and The Artful Dodger then writes special header data to the back-up disk and the back-up then works. No need for nibblers or parameters! The Artful Dodger **£9.95**

TOP SECRET STUFF VOLUME 1

A very powerful collection of twenty nine utilities used by the creators and master programmers of Utilities Unlimited. The first of many volumes is now ready. Included are programs which make the drive head run three times faster, stop the drive rattle, write protect a disk, unwrite protect a disk, rebuild a short new format, two DOS wedges, a disk matcher, smooth screen scroll, boot maker, Koala picture view and dump utility, a powerful monitor tester, a computer RAM test, plus much more. Every utility is ready to use. Top Secret Stuff Vol 1 **£6.95**

EXCLUSIVE DISTRIBUTOR

FSSL
FINANCIAL · SYSTEMS · SOFTWARE · LIMITED

HOW TO ORDER

FSSL, Defford Road, Pershore, Worc. WR10 1AZ

Telesales (0386) 553153

Open Mon—Fri, 9.30—6.00. Fax (0386) 556555
Enquiries Mon—Fri 3.00—5.30 (0386) 553222

Mail Order with Cheque, Postal Order or Access/Visa details. UK orders add £1.45 for postage. Overseas orders add £3.75 for Airmail. Please allow 28 days for delivery.
Cheques/Orders payable to FSSL.
Government & Eduction Orders welcome. All goods subject to availability. E & OE.

■ UTILITIES UNLIMITED ■ UTILITIES UNLIMITED ■ UTILITIES UNLIMITED ■

BASICS OF basic

A series of Basic tutorials designed to make the beginner an expert

JOHN SIMPSON

We have slowly introduced you to some of the more familiar KEYWORDS that make up the instruction set for COMMODORE BASIC V2. This month we take another step forward.

DATA AND READ

So far we have entered data at the keyboard, either by emplacing it within the program itself, such as Z = 10, or by using input commands, such as INPUT "A NUMBER"; NUM.

Now we are going to look at a way in which we can place data within the program itself and store it as DATA statements. Then we can use the READ statement to 'read' and to place the data in the appropriate places in the program. Most commonly data is stored either at the end of a program, between segments of a program, or at the very beginning of the program.

Here is a very simple program which will READ the data and then print it to the screen:

```
10 READ A$
20 PRINT A$
30 DATA JOHN SIMPSO
or
10 DATA JOHN SIMPSON
20 READ A$
30 PRINT A$
```

If you type this in, replacing my name with your own if you like, and then RUN it you will see that it

will print whatever string is held in the data.
We are not confined to strings. We can use numbers and manipulate them after we have read them.

```
10 READ N
20 PRINT N / 10
30 DATA 100
```

Run this and the result will be the printing of 10 (100 divided by 10).
We are not confined to just one piece of data either. We simply produce a loop which will cover all the data:

```
10 FOR X = 0 TO 9
20 READ D(X)
30 PRINT D(X)
40 NEXT X
50 DATA 10,20,30,40,50,60,70,80,90,100
```

LINE 10 Sets up a FOR...NEXT LOOP

LINE 20 Reads the data item and stores it in an undimensioned array D(X)

LINE 30 Prints the data value in that element determined by X

LINE 40 Iterates the loop

LINE 50 Holds the data to fill the array.or

```
5 DIM AA(13)
10 C = 0
20 READ D
30 IF D = -1 THEN END
40 AA(C)=D
50 PRINT AA(C)
60 C = C + 1
70 GOTO 20
80 DATA 1,2,3,4,5,6,7,8,9,10,11,12,13,-1
```

LINE 5 Dimensions a single column array 13 elements long.

LINE 10 Sets up a subscript counter variable C to zero.

LINE 20 Reads the data.

LINE 30 Places a 'trap' on the data - it tests each piece of data and if it equals -1 then the program terminates.

LINE 40 Places the current data item into the array of AA(C).

LINE 60 Increments the counter C.

LINE 70 Unconditionally branches the the program back to line 20 to read the next data item.

LINE 80 Holds the data to fill the array.

The first demonstration used a FOR...NEXT loop with which to fill the array and to determine the length of the data statements. The second example used a different iteration method by using, and incrementing, a counter. The point of the second example shows quite clearly that when we read data, we do not need a pointer to each data item; this is done internally for us. We can observe this from the fact that line 20 was simply, READ D. In other words, as soon as one item has been read an internal pointer will automatically select the next item to read. However, if you directed a read at data, and all of the data had been read, this would generate an error message:

?OUT OF DATA ERROR IN <line number>

line number being the line containing the READ statement.

To overcome this Line 30 set up a 'trap' which checks the data until it finds the data specified in the trap, in the above situation -1. As soon as it read this then the last data item having been read the program terminated. In the first example we used the FOR...NEXT loop which meant we needed to know precisely how many data items there are. In the second example we didn't. This is very useful when you may have very long data tables.

You can use as many Lines as necessary for your data, so long as each line starts with the DATA statement after the line number. Each item of data requires a comma between itself and the next one, except for the very last one on each line.

E.G:

```
100 DATA 100,200,234,123,573,298,236,981,
215,554,222,317,123,231,550,392,365,176
110 DATA 999,436,234,123,900,354,234,869,
942,123,456,872,905,881,345,288
```

Probably the most common syntax error in data statements is accidently using a stop (.) instead of a comma (,).

```
MIXING READS
10 READ A$
20 READ B$
30 READ C
40 PRINT A$,B$,C
100 DATA "JOHN","JENNI",2

RUN
JOHN      JENNI      2
READY.
```

From the above example you can see that you are able to mix string and non-strings into the data. What is important is the actual read statements, i.e., if you read a string you must use the dollar sign posttix, A$, B$ and a non-string or number is, as usual without a prefix, C.

When the computer encounters the first READ statement in the program it will read the first item of data, then consecutively through the data until it is instructed to stop reading, either at the termination of a loop or a counter. If at a later stage in the program you require the READ of more data items then the computer will continue READing from where it left off. You cannot read earlier data! There is a method, once you have READ all the data to RESTORE the internal pointer to the beginning of the data once again - however I will be dealing with this at a later stage in the series. Just remember that data is READ from beginning to end, for example, you may have three separate sections of data for three separate events. You can read them at any stage in the program, so long as they are consecutive to each other.

SUMMARY FOR PART 3

1.. We discovered the use of REMark statements, and the fact that when the computer interprets the code and it meets with the REM statement it ignores it and anything following it on that line. We only use this for reminders and remarks. Usually, once the program is finished, we save a version of the program with all the *REMs* removed to speed up processor time and shorten memory.

2.. We also had another glance at some of the screen editor's facilities. Namely how to overwrite commands on a line, delete a line, duplicate lines with common features, and how to incorporate a screen clear within a print statement by using a shift key and the CLR/HOME key together.

3.. This months lessons took us into the world of Arrays, both numerical and string types. We saw how easily we are able to create lists by using one dimensional arrays, and tables using multi-dimensional arrays.

4 . We discovered how simple it is to access the various elements in the array with the use of subscripts within loops. Later I will demonstrate how we can create and use arrays of subscripts(!)

5.. We plunged into our first glance at DATA statements and how to READ them and at the same time place them into an array. We saw that the computer reads the data in a consecutive manner using an internal pointer to each data item, and that once read it cannot be read again (although this is not strictly true, as we will discover when we have advanced further into the series). Soon we will discover how we are able to locate them anywhere in memory, even into those secret places which most people think are alien to the basic language.

In Part 4. Among other things we shall be taking a much closer look at formatting our text onto the screen, and using full colour, as well as the standard Commodore graphic characters with which to enhance our screen displays.

Until then, bye...

MULTIPLEXOR

Another method of obtaining more than 8 sprites is investigated RICHARD LITTLE

Various issues of CDU have contained SPRITE HANDLING routines, and some of these have given the user the ability to display more than eight sprites at one time. All however, have worked on the same principle. This method is fairly well documented, but here is a brief summary.

The uppermost eight sprites are positioned on the screen. An interrupt vector is set up so that once the raster scan (or electron gun in the T.V.) has passed the sprites and drawn them on the screen, the program moves all eight sprites down to the next eight positions. Typically this method allows three or four horizontal zones on the screen with a finite distance between them due to the processor time required to reposition the sprites into the next zone. Figure 1 shows this arrangement on the screen.

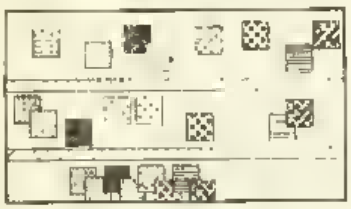


FIGURE 1

DRAWBACKS

There are several drawbacks to this fairly simple approach to MULTIPLEXING.

A) It proves difficult to move a sprite from one zone VERTICALLY into the next.
B) Large 'end of level' type aliens cannot be constructed as there must be a clear divide between each zone of sprites.
C) Games writing is restricted in that the positions of the zone dividers are awkward to adjust during play. No provision is made for the machine to work out the positions for itself.

LARGE ALIENS

How then, do many of the commercial games today manage to produce large aliens, often comprised of twenty or more sprites solidly fused together in one big lump? Well, this is how it is done. Please refer to Figure 2.
Let us say that we want our large alien to be comprised of sixteen sprites. As before, the machine places the uppermost eight sprites on screen. When the raster scan reaches position X the computer interrupts whatever it is doing and moves sprites 1 to 4 into positions 9 to 12. This allows the raster to proceed to position Y whereupon it moves sprites 5 to 8 into positions 13 to 16. This 'leap frogging' action can be continued down the screen to produce very large aliens. So what are the limitations of this method? As mentioned before it takes a finite time to move a set of sprites. This time can be translated into a vertical distance travelled on screen by the raster during the time in question. If the time taken to move eight sprites, redefine their pointers and change their colours is equivalent to a vertical distance of say 10 pixels, then the limitation is that nowhere on screen must we be able to draw a horizontal line 10 pixels wide through more than eight sprites. This sounds complex already, but consider the problems imposed by having a large alien on screen plus perhaps two sprites under human control.

MULTIPLEXOR TO THE RESCUE

The utility that I present here is designed to do all this thinking for you and make 24 sprites available on screen even to the Basic programmer. The routine is a refined version of the one which I wrote for the game 'PLAGUE', which was featured on the CDU MARCH 1990 disk. It is capable of dynamically controlling all of the interrupts required to show 24 sprites on screen, even as they whirl around AND cross over each other.

HOW IT ALL WORKS

First the computer must decide the order of the sprites in ascending vertical coordinates. Thus the sprites must be sorted. Extensive tests were done to find the fastest sort routine for the job. For just a few sprites a BUBBLE sort is adequate, however as the total approaches the maximum of 24 sprites such a sort takes up all the available processor time. The algorithm I settled for is a little known sort called the SHELL-METZNER sort. Derived from the better known SHELL sort it is lightning with long lists and niters by far the best average response time.

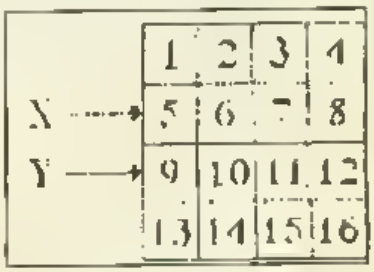


FIGURE 2

Once sorted the interrupt positions are determined by the PLEXOR routine and stored for use by the SPRITE MANIPULATOR. The manipulator is entirely interrupt driven and will display all sprites even as Basic is running.

MEMORY USAGE

Below are given the memory addresses which control the routine. Everything is stored in a small space from address $C000 (49152) onwards.

COORDINATES	(48 bytes)	$C000-$C02F (49152-49199)
COLOURS	(24 bytes)	$C030-$C047 (49200-49223)
POINTERS	(24 bytes)	$C048-$C05E (49224-49247)
No. OF SPRITES	(1 byte)	$C060 (49248)
BORDER COLOUR (during intl	(1byte)	$C08B (49291)
INITIALISE		$C2B0 (49840)
SORT		$C1EE (49646)

HOW TO USE PLEXOR

Once loaded (load"Plexor",8 or select from menu), the routine is activated by calling the INITIALISE routine (SYS49840). Sprite coordinates are stored alternate X then Y just as normal except you now have 24 rather than 8. The ninth bit of each X coordinate is stored as the most significant bit of the relevant COLOUR byte. You must define how many sprites you want to be active (0-23) in 'NUMBER OF SPRITES'.

For example, to place a RED sprite at coordinates 270,150 with a definition pointer of 128 you would,

```
POKE 49152,14
POKE 49153,150
POKE 49200,130
POKE 49224,128
POKE 49248,1
```

Then call the sort routine (SYS49646). Once the sprites are sorted the computer will wait until the PLEXOR signals that it is ready to accept the new positions. The interrupts are calculated and control is passed back to the users program. The sort routine need only be called when sprite positions have changed. The routines use their own workspaces so altering a coordinate etc will have no effect until the sort routine is called.

FINALLY

One other feature built in is the ability to sense when it is not possible, due to their positions, to display the sprites all at once. In such cases the PLEXOR introduces a controlled 'FLICKER', then sprites are displayed on alternate screen scans rather than not at all as in some 'commercial' games.
Included on the disk are two demos. The first is written in Basic to allow you to see how it is all done. It features a large alien and a small player controlled ship. Move the ship up and down using a joystick in Port 2. Interrupts can be viewed by pressing the fire button. Notice how the position of the interrupts depend on the position of the players ship. The second demo is written in machine code and shows the sort of effect you can have for title screens etc using this routine and a little knowledge of machine code for that extra speed. I hope you enjoy using this routine and that I have explained its use clearly.

HIRES CONVERTER

Logo making is given a helping hand SIMON COLLIS

I have a problem with designing logos. Whenever I attempt to design any large logo with a character designer, I always seem to look squared, not rounded, and monotonously colourless - not to mention flat. With a hires screen editor, however, I can do a lot better (although my graphical expertise does not make this much better). They say a bad workman always blames his tools, but I'm sure Michaelangelo would have complained if all he had to paint the ceiling of the Sistine Chapel was two old 4" paintbrushes and half a can of ageing Dulux.
But I still had a problem. While I wanted to design my logos with a hires screen designer, in the contexts I wanted them for, it was vital they be made from a character set. I was stuck, until I conceived the answer - write a hires screen to character screen conversion program. And so, I did.
Of course, it didn't just remain a hires picture converter - it also converts character screens to hires ones, converts pictures from one file format to another, and so forth. But enough of the introduction, what of the program?

GETTING STARTED

After loading the program (which can be accomplished by using the CDU menu, or typing 'LOAD "HIRES CONVERTER",8' and then typing 'RUN' when the prompt appears again) you will be presented with the following menu.

LOAD PICTURE
SAVE PICTURE
SHOW PICTURE
SELECT HIRES PICTURE FORMAT

LOAD SCREEN
SAVE SCREEN
LOAD CHARACTER SET
SAVE CHARACTER SET
SHOW SCREEN AND CHARACTER SET

CONVERT PICTURE INTO CHARACTER SCREEN
CONVERT CHARACTER SCREEN INTO PICTURE

DOS COMMANDS
CONVERTER COLOUR SCHEME
EXIT PROGRAM

COMMAND SUMMARY

LOAD PICTURE loads a picture, in the current picture format, from drive 8.

SAVE PICTURE saves a picture, in the current picture format, to drive 8. Note that the load and save formats need not be the same, but the "SAVE PICTURE" option does not save any machine code with formats which require it (such as "PAINT MAGIC" or "CDU PAINT" - to copy the code myself would have been a breach of copyright.)

SHOW PICTURE shows the hires picture currently residing in memory. Pressing "RUN/STOP" or "RETURN" will return you to the main menu, "M" swops between monochrome and multicolour modes, and "1" and "2" change the background and border colours.

SELECT HIRES PICTURE FORMAT presents a menu of picture formats (of which the default is "ADVANCED ART STUDIO") for picture loading and saving. Note that monochrome and multicolour picture formats are listed separately, and are denoted by the suffixes (HR) and (MC) respectively. Where no suffixes occur, assume the format to be multicolour only.

LOAD SCREEN and **SAVE SCREEN** allows you to save and load screens to and from drive 8. Screens are saved directly, with the restart address being set to $2000. Screens may be loaded with a restart address pointing to anywhere in memory. The file format, for those who want it, is as follows:

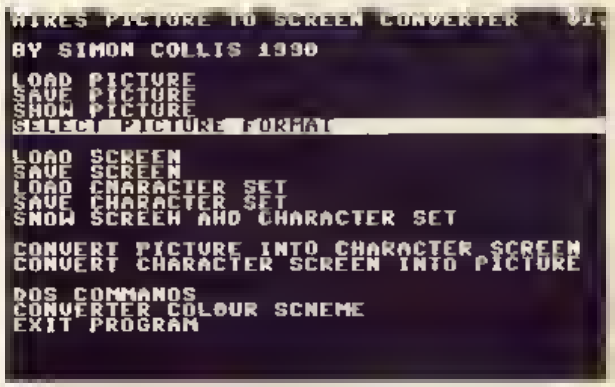
POSITION	LENGTH	DESCRIPTION
+$0000	$0002 bytes	Reload address (not applicable if loaded through kernal LOAD routine)
+$0002	$03E8	Screen character data
+$03EA	$0018	Unused
+$0402	$03E8	Colour data to be transferred to $D800
+$07EA	$0018	Unused
+$0802	-	End of file

LOAD CHARACTER SET and **SAVE CHARACTER SET** allow you to save and load character sets to and from drive 8. The reload address saved with the file is $2800. The file format is:

POSITION	LENGTH	DESCRIPTION
+$0000	$0002 bytes	Reload address (not applicable if loaded
+$0002	$0008	Data for character $00 (character 0)
+$000A	$0008	Data for character $01 (character 1)
	.	.
	.	.
.	.	.
+$07F2	$0008	Data for character $EE (character 254)
+$07FA	$0008	Data for character $FF (character 255)
+$0802	-	End of file

SHOW SCREEN AND CHARACTER SET displays the screen and character set currently held in memory. The same keys can be used here as in the "SHOW PICTURE" menu option.

CONVERT PICTURE INTO CHARACTER SCREEN converts the picture currently in memory into a character screen. With a monochrome picture, it will use its discretion to try and convert as much of the picture as it can into a character screen. With a multicolour screen, it will look at the pattern set in the "CONVERTER COLOUR SCHEME" option, and attempt to convert any pixels of that colour to the colour specified in this option. Note, therefore, that the same amount of colours are not so readily available on a character screen as are on a hires screen. Also be aware that the hires screen can contain up to 1,000 different 8x8 cells, and a character set can only contain 256, and so, at times, the conversion may fail, especially if there is a great amount of detail in the picture.



CONVERT CHARACTER SCREEN INTO PICTURE does the opposite conversion - takes a screen of characters and converts them into a picture. Useful for amending logos already drawn. There may, of course, be other uses, but I haven't found them yet.

DOS COMMANDS will print an "@" prompt, and at this point, standard C8M-DOS commands (as summarised on page 51 of the VC 1541 user's guide) can be entered, and the response will be displayed. If a "$" is entered, it will not be sent to the command channel (which would cause the response "31,SYNTAX ERROR,00,00") but will display the directory.

CONVERTER COLOUR SCHEME displays all the sixteen available colours, along with the current background and multicolours. Pressing the numbers associated with the 16 colours will alter the way in which pixels of that colour, when met, are dealt with. The 128 mode, when enabled, will blank the screen and enable 2MHz mode during conversion; when disabled, it will show the process of conversion as it happens, but be warned - the colours shown during this conversion are not accurate. Use the "SHOW SCREEN AND CHARACTER SET" option from the menu to check the colours.

EXIT PROGRAM will return to C8M BASIC. To restart the program, type SYS 3698. After this procedure (or a manual reset) a small section of the picture will be corrupted.

Finally, enjoy this program. If you have any comments, queries, or suggestions, (or extra picture formats) contact me through the magazine.

SCREEN DESIGNER/ COMPILER V2

The program originally published in SEPTEMBER 1990 gets a boost

ALAN WARRINER

Since this utility was first published last SEPTEMBER, I have made one or two modifications which make the program that much better to use. The main differences are within the SCREEN COMPILER/DESIGNER + BASIC COMPILER section (See below)

SCREEN COMPILER (address 52000-52775)

This program copies the current lo-res screen being displayed and creates a stand alone machine code program which will re-create the screen. The program automatically detects the video bank, screen address, character set pointer, colours and colour mode being used and restores these values when the screen is re-created.
The compiled code uses ten zero page addresses from 165 to 174 inclusive. To compile a screen, print the screen, then sys 52000,AD: where AD is the address at which the code is to be compiled, the program will return the end address of the code. To re-create the screen simply enter sys AD.
The screen data is not just stored as a block of data, but is compressed according to the following method The program searches from the start of the screen memory until the first character which is not a SPACE (#32) is found, this is the start of the screen. It then searches from the end of the screen, backwards, in the same way and marks this as the end of the screen. The screen data is then read in, compressed and stored. When the screen is re-created then screen is cleared and the video parameters are set,the data is then read and interpreted as follows:-
If the data value is less than 128 then it is OR'ed with a variable (REVELAG), which determines if reverse video is on or off, and stored to the screen, the current colour variable (CURCOL) is stored in the corresponding colour memory

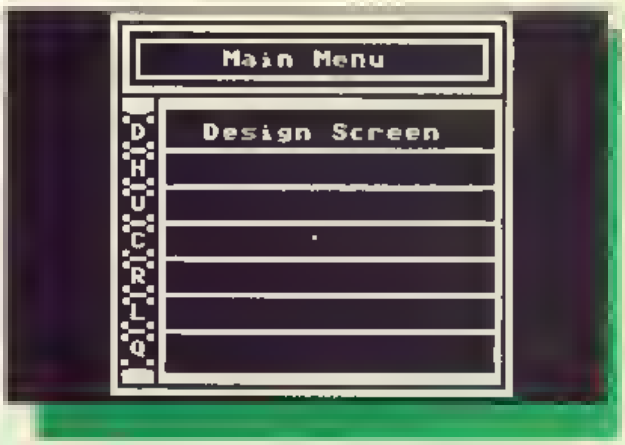
If the data value is greater than 127 then it is deciphered thus:-

If bit 4 is set then bits 0 to 3 hold the new colour value (CURCOL)
If bit 5 is set then REVFLAG is flipped, i.e. reverse on becomes reverse off and vice-versa
If bit 6 is set then there are a number of characters of the same value to be repeated, in which case, the next data byte represents the number of characters to be done, and, the byte after that holds the character value to be used.

More than one bit may be set at a time

If bits 0 - 6 are zero then this is the last byte and the routine exits

The program "compiler 52000" on the disk is a basic loader for "compiler code", which is the program itself

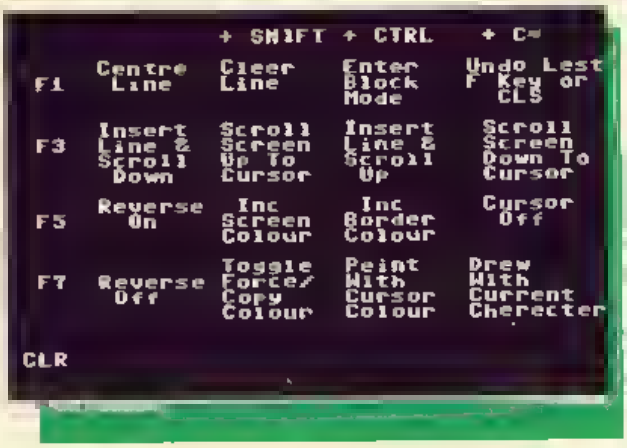


SCREEN COMPILER/DESIGNER + BASIC COMPILER

Address 16384 - 29798. The main change is the ability to now compile BASIC code as well as machine code from within the program, the Basic code may be in the form of a single subroutine to re-create the designed screen, or, a series of subroutines for different screens, or, subroutines appended on to the end of existing Basic programs.

When the compile option is first selected from the main menu you are given the choice of Basic or Machine code, whichever is selected then becomes the default path for compiling i.e. when selecting

the compile option you will be immediately taken to the choice you have just made. If you wish to be given the Basic/Machine code choice again either press SHIFT+C at the main menu or SHIFT+RETURN on the compile option. If the machine code option is taken then proceed as with the original program. If the Basic option is used and this is the first basic screen and no basic program has been loaded then you will be prompted for a line number, this will he the first line of the program the line number must be in the range 0 - 60000. When the code has been compiled you may then save the code or design more screens and append them to each other. The next time you compile a screen or if you have loaded a basic program into memory then you will be asked if you wish to create a New program or to Append the code onto the existing program. New will prompt you for a line number as before but Append will either, use the next line number if appending to a compiled program, or default to line number 50000 if appending to a loaded program, in both cases the first line number will be displayed. You may append as many subroutines as you like to the programs until the memory is full (highest address 16384) When memory is full you may still save the code up to the last complete subroutine. Basic code cannot be recalled by the program.



DESIGN SCREEN

In the designer you may type in characters and change colours in the normal way, but in addition the function keys will give the following effects.

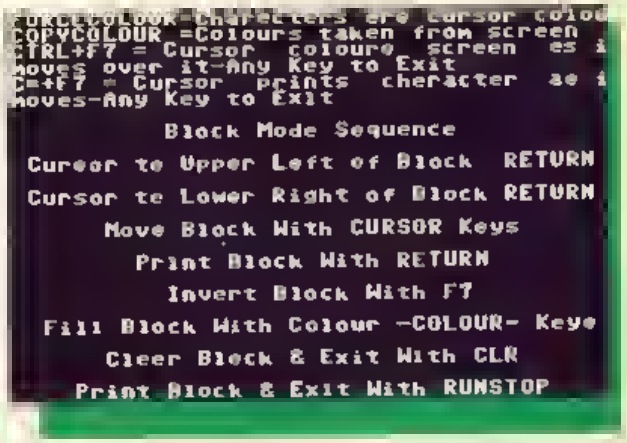
F1 - Centre text on cursor line.
F2 - Clear cursor line.
LOGO+F1 - Undo last clear screen or function key.
CTRL+F1 - Enter block mode. In block mode you must follow the following sequence.

Move cursor to upper left corner of block and press RETURN

Move cursor to lower right corner of block and press RETURN.

The block is now defined and may be moved around the screen with the cursor keys, in addition:-

RETURN - will print the block.
Any COLOUR key will fill the block with that colour.
F7 - reverses all of the characters in the block
RUNSTOP - will print the block and exit.
CLR - clears the block and exits.



CONTINUING FUNCTION KEY DEFINITIONS

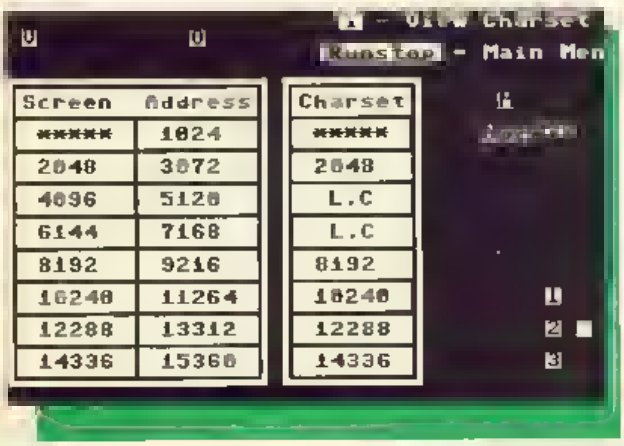
F3 - Insert a blank line and scroll screen down.
F4 - Scroll screen up to cursor.
CTRL+F3 - Insert a blank line and scroll screen up
LOGO+F3 - Scroll screen down to cursor
F5 - Reverse on.
F6 - Increment screen colour.
CTRL+F5 - Increment border colour. (SHIFT+CTRL+F5 increments both screen and border)
LOGO+F5 - Switch cursor off, any key cursor on
F7 - Reverse off.
F8 - Switch Forcecolour/Copycolour mode. Forcecolour is the normal typing mode in which the character printed is the colour of the cursor. In copycolour mode the character is the colour of the existing character on the screen, unless, that colour is the background colour in which case the last, selected, colour is used.
CTRL+F7 - Paint with cursor colour. In this mode as the cursor moves over the screen it colours the characters it moves over, any key, other than the cursor keys, exits this mode.
LOGO+F7 - Draw with current character. In this mode the character which is under the cursor when the mode is selected is printed as the cursor moves over the screen, the colour is determined by

Forcecolour/Copycolour.

To exit the designer press RUNSTOP to return to the main menu, or, SHIFT+RUNSTOP to call up HELP screens.

HELP

This will call up the designer help screens RUNSTOP returns to main menu any other key calls second help screen.



VIDEO PARAMETERS

On this screen you may select the video bank, screen address, character set pointer, colour mode and colours used by the designer and compiled code. You may also view the selected character set. Certain addresses are not available as they are used by the program. RUNSTOP to return to main menu.

COMPILE SCREEN

On this screen you will be prompted to enter the address at which you want the designer screen code to be compiled, if you have previously compiled a screen you will be informed of the next available address to allow you to create consecutive blocks of code, pressing RETURN without entering anything will automatically compile at this address. You will not be allowed to compile code which overwrites the program, this is not a problem as you can compile the code to a different address and then use the separate screen compiler to move it to your desired location. Once code is compiled you have the option to save the code in which case you will be prompted for a filename. Code may be compiled and recalled from under the I/O area and under the Roms, but, only code under the Basic Rom (40960-49151) may be saved. The program uses the area under the Kernal Rom (57344-65535) as a workspace and so any code compiled there may be corrupted.

RECALL SCREEN

This allows you to put a previously compiled screen into the designer, also, all of the video parameters are read from this code. If no code is found at the address you specify then you will be asked if you wish to search for code, if you do then the program will search from that address until valid compiler code is found which is then loaded into the designer, you may then exit or continue to search.

LOAD/DIRECTORY

Selecting this option allows you to view the disk directory and load from it. You are not allowed to load code whose start address is in the program memory space. When code is loaded the load start and end addresses are displayed, and, if the code is a compiler screen it is loaded into the designer and a red asterisk (*) is displayed to inform you this has happened.

QUIT

Reset computer. Re-enter program with sys 16384.

The programs on the disk are "designer v2" which is a loader for the main program which is "scad"

ART ON THE 64

The proven and profound pleasures of paint package power

JENNI SIMPSON

As as artist and a great lover of all things 'natural', I was, as you can well imagine, rather disconcerted, when a friend eagerly suggested that I try my hand at using 'The Advanced Art Studio' Graphic/Paint Package, for the Commodore 64 Computer.

"You'll love it", my buddy assured me, with an abundance of enthusiasm. But I was not so convinced. For up until now, I had believed that Computers and the World of Art were completely incompatible. In fact, for me, they were simply a 'contradiction in terms', rather like trying to mix oil with water! And anyway, apart from this 'major' criticism, I was used to having at my disposal vast arrays of different colours and textures of a more traditional nature, and my mind simply boggled at the very thought of having to work with such a complicated medium that was so 'technical and alien' to all that I knew.

THE TWILIGHT ZONE

However, not being a stick-in-the-mud, fainthearted, or one to dismiss something before I had really given it a fair 'crack of the whip', I threw caution to the wind, and apprehensively embarked upon the unfamiliar, and, what I thought would turn out to be, an arduous journey. HOW WRONG ONE CAN BE! For after being shown to the cockpit, supplied with a steaming cup of coffee, and then briefed with a few simple and helpful instructions, I began my enchanted journey into the 'twilight zone'.

HOW I BECAME A SECRET AGENT

First of all, I had to hold down the right button on the mouse, whilst I switched on the '64. Easy enough! I then loaded the program from the disk drive, having to wait for two or three minutes for it to do so. No problem! Instructions then appeared upon the screen asking me to find page 20 in the user manual, and to type in the space provided, the 9th word on the 6th line (this was an exciting moment for me, because I felt as if I were a Secret Agent breaking into National Security).

The code word typed in, I then watched with amazement as immediately a Double Menu Bar, with 11 menu options, appeared like magic upon the top two lines of the screen. I later learnt that these lines did not have to be wasted but could, in fact, still be used by scrolling the screen to reveal them.

By now, I was completely enthralled and thoroughly enjoying myself, following the simple instructions, and waiting expectantly for further developments. 'Brill stuff'!

MEGALOMANIA TAKES OVER

I then discovered, much to my profound pleasure, that when I moved the mouse, there was a pointer on the screen, that I had sole control over. Strains of megalomania began to seep into my consciousness as I realized the true potential of this marvellous 'new toy'

So simple to use, I found I could, by means of this 'mechanical rodent', select and click onto any option that I desired, which then facilitated a pull-down sub-menu window with additional options for that particular function.

The 11 main options (namely, Print, Colours, Fill, Paint, Text, Undo, File, Windows, Magnify, Shapes and Misc), were, I felt, on the whole practical and very versatile. Choice of colour could be selected (would you believe it?) from the colour menu, although I must admit I was rather disappointed that there were not more colours to choose from (there are 16 in all, the usual C64 standard colour set).

STEVE 'INTERESTING' DAVIS

Having said this, though, the package is in the normal multi-coloured mode. Fortunately, this does allow for four separate colours to share an eight by eight pixel character space, which then gives three ink colours plus a background colour. Pretty impressive. Although I was able to draw graphics using several different sets of ink colours, I found that occasionally, if one set happened to overwrite another, then this could lead to a clash of colour priority. However, in certain circumstances, the overall picture could be enhanced, lending a Steve 'interesting' Davis blurred effect to some locations ! Nevertheless, these are only minor quibbles, for there were so many 'things' for me to do, I almost forgot to drink my coffee, and believe me, that is most unusual!

FEELING SHAPELY

If I felt in a 'Shape' mood, I could, within an instant, select from my Shapes Menu. I found I could easily draw Straight lines (single or connected), Rectangles, Triangles, Circles (two types), Rays, and could even select for Individual Points. These options having an elastic line mode which could be either switched on or off as desired.

The shapes could be filled with colour either solid or patterned, and speaking of patterned, I should just mention, that even these can be edited and saved out, giving you an almost limitless range of patterned textures to choose from.

PEN, BRUSH, AND SPRAY THOSE BLUES AWAY

Of course you don't need to use the shapes, you can select the Brush menu which gives access to a variety of brushes, single or many coloured. Again, like the patterned fill, there is a brush editor allowing you to create an almost limitless number of brush patterns. I found the brushes extremely useful for painting bold areas of my creations.

For 'graffiti maniacs' there is a spray-can with an adjustable nozzle for intensity, which can be used for fine to dense spraying. Using several sprays with different colours each time gives some incredibly textured pieces of art from subtle tree foliage to intense starry night-time sky displays.

There is also a range of 'pens' available which can be used for free-hand drawing and cross-hatching. The range is from a couple of pixels thickness to about six or eight, and angled for best results. Marvellous stuff!

MY WORD! WOT A BIGGA PICTURE!

Any part of the screen can be magnified 2, 4, or even 8 times, with or without a grid overlay, to enable you to produce extremely fine detail, such as eyes, nose and mouth areas of a face, which, when reduced back to normal size, occupy only a small section of the screen.

In addition to all of this, is the added advantage of delivering text to your creation. The characters can consist of nine possible point-sizes, either bold or italic, which can then be custom designed, if so wished, using the built-in font editor.

PORTABLE WINDOWS

The Window's option is excellent. Here you are able to create a small window, perhaps a section of brick, a window, a door, or in fact anything you wish. These can then be manipulated with rotation, or inversion, etc., and, if so desired, saved out. Later, during the creation of your picture, you are able to recall the windows, and locate them anywhere you like within your art work. What more could you ask for!

THANK GOO FOR TINY RODENTS!

Using my "cheezee lovin' friend", instead of the usual pencils, pens, brushes, crayons, etc, was, I must say, at first, very daunting. But after a while I soon got the feel of it, and was eventually creating all sorts of artistic goodies with a variety of pens and user-definable brushes.

USE A RUBBER!

The facility of erasure is, of course, fully operable, with a simulated rubber, an undo option, or even a complete screen clear. So there is no need for concern, for if your creation does go radically wrong, it can easily be amended.

BEYOND THE FUTURE

All in all, my pre-conceived, ignorant prejudices concerning the Computer and the World of Art, have been totally shattered. I can now see the limitless possibilities of imaginings yet to come. For this, I believe, is the dawn of the age of computing, and the only limits there are, are those within the mind. And now, with the advent of 16 and 32-bit machines, like the Amiga, the Atari, and the PCs, plus the increasing clarity of definition and the vast mega-million range of colours, the future is, I feel, becoming truly phenomenal.

LOADS 'N' LOADSA FUN

So why don't you get yourself a paint package? There are so many to choose from, for example 'Artist 64', 'Blazing Paddles', 'The Image System', 'Koala Painter' and 'Vidcom 64', just to mention but a few. You really have got nothing to loose, and I feel sure that you'll have loads and loads of fun for hours and hours on end!

I am absolutely convinced that inside each of us lies the potential to become a budding RENOIR, CONSTABLE, REMBRANDT or even DALI

So go on, spoil yourself.

The cover picture of this month's magazine serves as a good illustration of what can be achieved. The world's your whelk!

HAPPY DOODLIN

EASY WORKING PROGRAMS

A 3 in 1 office package gets a viewing ADRIAN MILLETT

In this day and age, a review of a set of WORDPROCESSOR, SPREADSHEET and DATABASE programs for the 64 is unlikely to excite the average magazine reader. Nevertheless, if the price weighs in at 5p under under 20 quid, you may be induced to take a second glance

THE PACKAGE

The packages are called "EASY WORKING WRITER", "EASY WORKING PLANNER" and "EASY WORKING FILER", they are from an American publisher called SPINNAKER SOFTWARE CORP. Your hard earned cash buys you 2 double sided disks, and not much else! You have to print the manual for these packages on your own printer by running a program called "MANUAL". To be fair, I can understand why this is done, since 19.95 does not allow much margin to cover a vast quantity of printed documentation. This idea seems to be getting more common these days, and I personally don't mind as long as the quality of the manual is good.

MINOR PROBLEM

However, when I came to actually trying to run these packages, I hit a serious problem. They would not load! I should expand on that:- LOAD"*",8,1 should invoke a 1541 TURBO-LOADER and run the software, but it simply refused to do so on my standard C128 (in 64 mode) and 1541 setup. I have figured out an alternative loading procedute, but that is not the point. 1541 turbo loaders are notoriously unreliable, and many people own drives like the 1570, 1571 and others that are even more problematic in this regard. In my view software houses should ALWAYS provide an alternative non-turbo method of loading their programs. I generally don't have problems with turbo-loaders, so I suspect this problem might be related to differences between American and English C64's. Here are the alternative loading methods;

LOAD"EW*",8,1 <RETURN> SYS2304 <RETURN> for EW Writer
LOAD"EB V*",8,1 <RETURN> SYS2304 <RETURN> for EW Filer
LOAD"EP*",8,1 <RETURN> SYS2304 <RETURN> for EW Planner

EASY WORKING WRITER

At first glance, I was quite impressed with EW.WRITER. After selecting 'EDIT' from the main pull-down menu, I was able to enter a short letter with a few seconds work. The editor behaved exactly as you would logically expect it to, with the usual selection of cursor controls, word wrapping within paragraphs and so on. Further assorted function and control keys provide extended cursor movement by word
page and a healthy selection of block delete, move and save functions The F1 key produces a pull down menu which allows you to insert a print formal command (ie: BOLD, UNDERLINE, NLQ, JUSTIEY and HEADER) at any point in your document. You can also select options to search or "search and replace" text, and there is a text preview facility, so you can see what the final printout will look like without wasting paper. The spelling checker built into this package is neat and easy to use, although fairly basic. When it finds a word it doesn't know it simply asks you to edit the word in question, without trying to actually suggest the proper spelling as some other wordprocessors do.
Once your text is in order you can, of course, save it to disk. The EW range of packages all add an 'extension' to the name of your file. The extension for a word processing document being ".LIS". This excellent idea, borrowed from IBM-MSDOS, really does help you to manage your files efficiently. To re-load a text file, you can select directory and use the cursor keys to choose the desired file - this is always a good feature.
A good range of printer options are provided. You can setup the printer device number and type, and define exactly what codes are needed for BOLD, NLQ and other printer modes. Unfortunately, the old user-port type printer interfaces are not catered for. Text may be printed direct, printed to a file or merged with another file (from, say, EW FILER) and then printed. There are further

options to set page length and to pause after each page for sheet-fed printers.
Unfortunately EW WRITER does have a problem, in the shape of a bug! When the cursor is at the end of a paragraph and you use the delete key to erase characters to the left, the cursor sometimes "catches" the character to the left without actually erasing it. This very annoying bug spoils what is otherwise a reasonable little wordprocessor

EASY WORKING FILER

EW FILER is a good example of one of those card-index type databases, nothing revolutionary, but it does the job. As with EW WRITER, I found I was able to knock up a simple database without having to refer to the manual. To kick off, you simply select the "NEW DATABASE" option, and specify the name, type, size and format of each field in your database. You must make one of these fields a 'KEY FIELD', which is the principle one you intend to use when accessing data at a later stage. A field can be specified as ALPHANUMERIC, NUMERIC or DATA, (MM-DD-YY American form!) with further controls on whether the field is CENTERED, LEFT/RIGHT JUSTIFIED, POUNDS/PENCE format, and of course the size of each field. Surprisingly the maximum size of any one field is only 25 characters, so that a large field like an address would need to be sub-divided into smaller fields like ROAD, TOWN, etc. The maximum combined size of all the fields put together is only 250 characters per record, with a maximum of only 10 fields, so I'm afraid that cross-reference index of WAR and PEACE you were planning is right out. Once the database has been specified, the program kindly checks the amount of free disk space, and calculates the maximum number of records you can fit on the disk. You may make this amount smaller if you wish.
After the file has been created, you can start ENTERING, EDITING, SEARCHING for and PRINTING data. The SEARCH facility for VIEWING or PRINTING data looks quite good on the face of it. PATTERN MATCHING, like in DOS, can be used when searching for ALPHA fields, and comparison operators (<,=,>) can be specified as well. For instance you should be able to search for all records whose name field begins with 'A' (=A*) and with an amount field of over ten quid (>=10.00). In practice, the matching for numeric amount fields doesn't seem to work! In the previous example, the search was yielding amounts BELOW 10.00 with the database I was experimenting on. So another irritating bug has crept through the net I'm afraid.
When printing, you are given quite a bit of flexibility in the way the data is formatted. Fields can be positioned almost anywhere, with as many records down the page as you wish. This flexibility is handy for printing on things like tractor-feed mailing labels, that often come in a variety of sizes.

EASY WORKING PLANNER

This SPREADSHEET program completes the now familiar standard trio of programs in "INTEGRATED" packages like these. EW PLANNER provides quite a large spreadsheet for you to work on, with 250 rows by 250 columns of data cells. In keeping with tradition, the cells are labelled from A1 (top left) to IP250 (bottom right), and each cell can contain either text (in quotes), numerical data or mathematical formulae that perform calculations based on the contents of other cells. The usual range of mathematical operators are provided for calculations, together with a selection of functions that can calculate the SUM, AVERAGE, MINIMUM or MAXIMUM of a range of cells. You are able to move the cursor freely from cell to cell, and enter or edit data at any location. A range of menu options are provided for SAVING, LOADING and PRINTING out the spreadsheets, operating in the same way as the EW WRITER menus. In addition, you can save and load work-sheets in a special universal "DIF" file format that can be used with other spreadsheet programs, a sort of VHS standard of the spreadsheet world.
I did find a small quirk when trying to print a simple spreadsheet I had set up for test purposes. The spreadsheet was in the form of an invoice for goods, with descriptions of items running down the left hand side, a value for each item down the right, and a grand total at the bottom of the right hand column. Now when you have a text field larger than the normal cell width the usual practice is to allow the text to overlay cells to the right of the field, which you should allow for by leaving some blank cells. In my example the description column was usually much larger than the normal cell width of 8 characters. With EW PLANNER I found that this worked fine on the screen, but when printed these large fields pushed over columns to the right, with the result that the "TOTALS" column didn't line up. Fortunately there is an option to change the width of all cells, so I was able to get round this problem in my example by setting an appropriate cell size.

IN CONCLUSION

The basic idea behind the "EASY WORKING.." programs is good.- someone has sat down and tried to design a set of programs that share a consistent user interface. The pull down menu system is well designed and easy to use, and many commands and control keys are common to all of the programs. I generally found I could start using them without reference to the manuals at all, and there is plenty of context-sensitive help for when you get stuck. The manuals are generally good, providing the necessary detail in a clear, logical manner
To be honest, I expect to find the odd bug (or "feature" as software publishers call them these days!) in any commercial package, particularly in packages for the C64. However, most of the problems I found here were not very deeply buried, and really should have been found if someone had tested the packages out properly.
It's a great shame to see such promising software spoilt by a few simple bugs.

PROGRAM: *EASY WORKING WRITER*
SUPPLIER: *E.S.S.L. Masons Rhyde, Defford Road, Pershore, WORCS*
PRICE: *19.95 (Disk Only)*

UDG SYSTEM.2

A comprehensive graphics package for program designers

JENNI

UDG SYSTEM.2 is a very sophisticated and extremely powerful character editor and screen designer. It has many novel features and is very user friendly making it the perfect application tool for programmers of all types - whether you're a beginner or a professional.

ALL IS REVEALED

From the power-up state there are four main modes, which are:

1. MAGSET.
2. CHARSET.
3. MAPEDIT.
4. DISK/TAPE OPS.

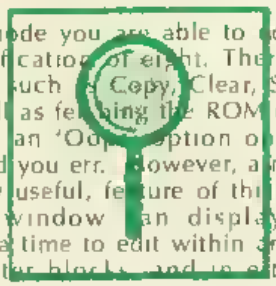
The screen is divided into four distinct areas, or windows. Window 1. is where the character set currently being worked upon is situated using a 32X8 viewing area. Next to this is window 2. an 8X8 area which is termed as ICONMENU. This displays two icons - the upper, and larger icon being the main one of the two. Below these two areas is window 3 , a command/status line (40X1), and below this is window 4. a 40X16 main editing area.

A movable arrow pointer is pointing to the upper icon in the IconMenu, and the symbol displayed is a magnifying glass.

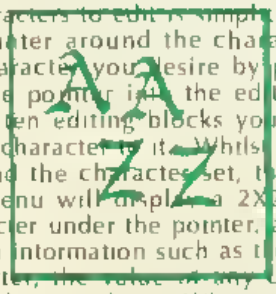
Apart from using the keyboard for disk/tape files and housekeeping, the whole of the utility uses the joystick in port two plus the occasional use of function keys F1/F7 (Keys can be used instead of the joystick, and sometimes I found this to be very useful, for example, when making minor adjustments to the pointer).

Whilst the pointer is 'locked' onto the IconMenu pushing it up/down will cycle the icons through the four modes stated earlier. Fire will activate the mode displayed by the icon. Once you have entered the selected mode the IconMenu now displays all the options available within that particular mode, again, by cycling through them and selecting the option you require by pressing the fire button.

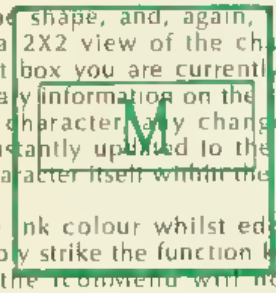
MAGSET MODE



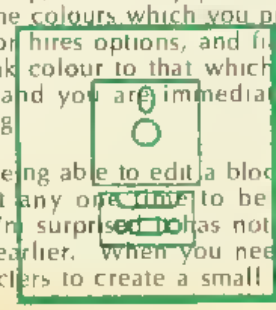
In MAGSET mode you are able to edit characters with a magnification of eight. There are all the usual options such as Copy, Clear, Scroll, Invert, Rotate, as well as fetching the ROM character set. There is also an 'Oops' option to most of the options should you err. However, almost unusual, and extremely useful, feature of this mode is that the editing window can display up to ten characters at a time to edit within an area of 5X2 (8X8) character blocks, and in either hires or multicolour mode.



Selecting characters to edit is simplicity itself, just move the pointer around the character set and 'grab' the character you desire by pressing fire, then move the pointer into the edit area, select which of the ten editing blocks you want to use and copy the character to it. Whilst the pointer is moving around the character set, the upper icon within IconMenu will display a 2X2 view of the current character under the pointer, and the status line gives you information such as the ASCII value of the character, the value of any character you might be holding, and etc. When you move the pointer into the edit window it changes to the usual box type shape, and, again, the IconMenu will display a 2X2 view of the character in the particular edit box you are currently within, plus all the necessary information on the status line. As you edit the character, any changes which are made are constantly updated to the 2x2 view, as well as the character itself within the character set



Changing the ink colour whilst editing is rather neat too. Simply strike the function key F7 and the icon within the IconMenu will now display a coloured tile. Up/down the joystick will cycle the tile through the colours which you preset from the multi-colour or hires options, and fire will change the current ink colour to that which the coloured tile displays and you are immediately delivered back to editing



I found that being able to edit a block of up to ten characters at any one time to be such a good feature that I'm surprised it has not been thought of and used earlier. When you need to combine several characters to create a small UDG, such as

a tree, or a brick, or a panel, or even a 2X2

character, this option is perfect for doing just that. No more editing one character then calling up the next, doing a small edit on that one and then having to call back the first, and so on. It can all be done in one go. Excellent, and time saving too.

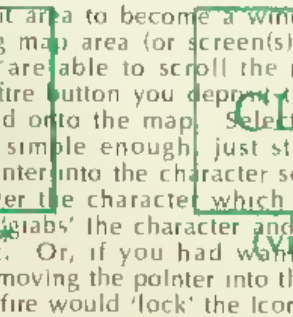
The lower Icon in the IconMenu is a switch to display or clear a grid network over the entire edit window. This is another useful facility to help with tricky graphics and alignment, although on the status line the x,y coordinates of the edit box you are currently within are displayed.

CHARSET MOOE

Selecting the CHARSET mode takes you into character editing directly upon the character set displayed within window 1. All the options from MAGSET are there but with some useful additions. I found this mode extremely useful for dealing with 'mirrored' designs. Let's say you are designing a box shape. In MAGSET you would simply design just one corner, then moving into CHARSET you can copy this shape to a further three characters (any at all, and simply done with the pointer), now selecting the rotate option you are able to rotate them until you have the four separate corners. Of course this is a straight-forward use of the option, but more intricate and clever designs can be achieved using methods such as this.

MAPEOIT MODE

MAPEDIT mode is where you can design your screen using the character set you have just created. The edit window now becomes the map edit window, although it has been reduced by one line to 40x15 this is because the status line has increased to two lines to cope with increased information. There are eight options to select from within this mode. You can move into EDIT. This forces the edit area to become a window in front of a scrolling map area (or screen(s)). Using the joystick you are able to scroll the map and by pressing the fire button you deposit the character currently held onto the map. Selecting another character is simple enough, just strike F7 and move the pointer into the character set, when the pointer is over the character which you require pressing fire 'grabs' the character and returns you back to edit. Or, if you had wanted another option, then moving the pointer into the IconMenu and pressing fire would 'lock' the IconMenu ready to be scrolled and a new option selected.



You are able to select a character with which to fill the entire map area. For example, the area may be filled with all sorts of garbage, so selecting a blank space will clear the map ready to be filled with your graphics. You can also select a character with which to fill the border around the map area. Again, if you have filled the map with blanks, a border pattern can help to identify the edges of the map, although the status line does show you the current x, y coordinates of your map

You are able to nominate exactly where you require the start of the map to be in memory, restricted to the area from location $4000 to $BFFF (32k), and you can coordinate the map into X, Y coordinates. For example you could set the map to be a 255x160 character oblong, starting at address $4000.

There is also a useful option called SNAPSHOT, this allows you to select an area (up to 8k), and deposit it elsewhere within your map. This is very useful to use where you may have several repeated graphic blocks, or simply for filling large areas of the map with the same character(s).

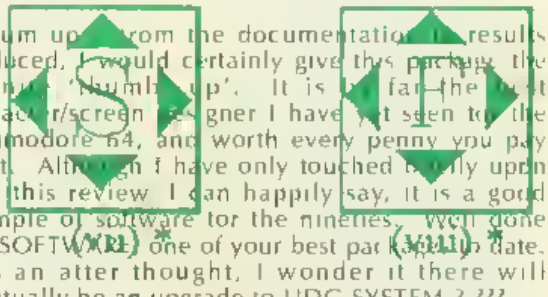
VIEWMAP MODE

VIEWMAP enables you to view the entire map using the standard screen (40X25). If the map is larger than the screen, then a fast eight-way scroll is activated from when you use the joystick

OISK/TAPE OPS MOOE

This mode is where i/o housekeeping is handled. Full save/load facility of character sets and maps, and for the disk user, all disk commands can be executed from a hi-lite bar pull down menu. The disk directory is loaded into tree memory space so that when viewing long directories you can scroll backwards or forwards through it

The documentation which comes with the package is very clear and precise, explaining in great, yet very simple, detail every option and aspect which is available.



To sum up from the documentation to results produced, I would certainly give this package the definite 'thumbs up'. It is by far the best character/screen designer I have yet seen for the Commodore 64, and worth every penny you pay for it. Although I have only touched briefly upon it in this review I can happily say, it is a good example of software for the nineties. Well done ESP SOFTWARE one of your best packages to date.

As an after thought, I wonder if there will eventually be an upgrade to UDG SYSTEM 3 ???

PROGRAM: UDG SYSTEM.2
SUPPLIERS: ESP SOFTWARE, 26 RIDGEWAY, BERKHAMSTED, HERTS, HP2 4LD
TELEPHONE: 0442 86631
PRICE: 12.95 (Disk Only)
RELEASE DATE: 1st JUNE 1991

TECHNO-INFO

The CDU mailbags get bigger and bigger - JASON FINCH wades through them

This month we have a bumper selection of letters for you to chew your way through. There are no less than fourteen queries ranging from questions about starting out on the C64 to ones about specific peripheral problems and special features of the C128. You will also find the new UPDATE section near the end that keeps you posted on any developments regarding past queries, and of course the TIP OE THE MONTH is here as well. Hopefully there will be something for everybody and so without further fuss, let's get the show on the road.

FOR YOUR EYES ONLY

I have the CDU disk from volume 4, issue 3 and I am having a couple of problems. I read the instructions for the encryption program and when I wanted to use it, I loaded up ENCRYPTER.SYS as detailed in the magazine. I then tried to load the program to be encrypted but I got an error - an OUT OE MEMORY error. I have tried various alternative ways of loading but I have had no success! I cannot load the program that I want encrypted. No way. Ok, so I load another program from the disk. I load SECURE and then I must load the program that I want protected. I do. What?? I see the message "Enter password" from the program ENCRYPTER I follow the instructions and now the program is encrypted but not protected. What now? Please help me because I am very confused as to what is happening.
Peter Wischhoff, The Netherlands.

Dear Peter,
I am quite confused as well to tell the truth because I can't work out what you did in between loading Encrypter and Secure. Unfortunately you don't tell me whether you reset the machine or just did a STOP/RESTORE combination. I will however endeavour to tell you why what happened did happen and then to tell you what you should do. First of all, the reason for the original error is due basically to pointers being changed due to the load address of the machine code. When you loaded Secure you must have forgotten the ,8,1 suffix after having reset or something which will have resulted in Secure being loaded to the normal address for BASIC programs. Your other BASIC program will have overwritten this and will have not produced an error because the code was not loaded to the correct area at 49152. Then, when you typed SYS 49152 to activate Secure, you will have activated the Encrypter code that was still in the memory. To enable you to use the programs properly you should load the ENCRYPTER.SYS program and then when the READY prompt reappears you should either type NEW to reset the pointers or enter SYS64738 to cold start the computer which also resets the pointers. Neither of these factors will wipe the encryption program out. So then you can load the BASIC program followed by SYS49152. The same procedure should be done when loading Secure - type NEW or SYS64738 after loading the code with the ,8,1 suffix.

128 SPECIFICS

Dear CDU,
As CDU is the only Commodore magazine here in Greece with serious articles, it is the only one that could provide someone to answer my following questions. I hope you are able to do so. Firstly, in multicolour graphic mode, two of the colours are kept in locations $1C00-$1EFE. Where is the third colour kept? Secondly, where are the data (of colour and characters) for the 80-column screen kept? And how does the memory map change when selecting the 80-column mode (if it changes)? If I want to create my own character set, I have to transfer the character set data from ROM to RAM. Which memory location do I have to update so that the VIC-II reads the data from the new area? (for C64 it is $D018 but I don't think so for C128). Is this location the same when using 80-column mode? I suppose that 80-column character set data is different from the 40-column. Where is it all stored? Next, how does the POINTER function work? For example if I declare A=10, then POINTER (A) = 1026, but this memory location doesn't seem to help me find where A is stored. Also, which zero-page locations must I not change and which can I use to store values? Could you recommend any books on programming sound with assembly on the C128 and any books for beginners on interfacing and any others on interrupts All my

questions refer to the C128 in its native mode.
George Thalassinos, Greece.

Dear George,
Phew - what a lot of questions! I don't mind of course for that is the reason that I am here. Your first question is a very good one and one for which I have always wondered the answer myself! I won't pretend that I know the answer because I don't unfortunately. However, one of the books mentioned later may be able to help you. The eighty column mode problems are very tricky. It is all controlled by the 8563 Video Chip, the VDC, a special chip in the 128 which can only be accessed through TWO locations. You send codes through one location and additional information for that code through the other. It is a very difficult concept to explain in this space and so I won't attempt to do the impossible. On the disk you will find four programs for you, filed as PROB1 with alphabetic suffixes (ie. PROB1A, PROB1B and so on). Quickly though, the VDC is 16K's worth of information and is laid out as follows: $0000-$07FF is the screen memory map. $0800-$0FFF is basically the colour memory, $2000-$3FFF is the character definition data. The two locations in the C128's memory for controlling it all are $D600 and $D601. There are a number of registers that you can access and by POKEing various values into $D600, followed by different ones to $D601 you can do such things as change the screen map, the character definitions and so on. It is all VERY complicated I assure you but the programs on the disk should demonstrate. The first two deal with redefined characters, in 40 and 80 column modes respectively, the third one deals with screen map manipulation in 80 column mode and the last is a complete BASIC program illustrating a few tricks that can be obtained with the 80 column screen and the VDC. Programs two and three are both BASIC programs that read in and run machine code. Everything is fully REMmed and disassemblies are provided in the programs where appropriate. I will perhaps supply an article on the VDC at a later date. With forty column character sets, the equivalent address to 53272 is 2604 (decimal). If you put the characters at 8192 onwards you must use POKE 2604,24. There is no equivalent for eighty columns - you just change the definitions. To use the POINTER function you must enter BANK1 before you PEEK the locations given by the function. If the value given is 1026 then the name is stored in locations 1024 and 1025 and the location is at 1026, 1027 and 1028. The info is in the upper set of 64K within the 128. (ie: 1026 corresponds to $10402). Page zero is used as follows. $00/$01 are special I/O control registers, $02-$D6 are used by BASIC and the operating system, $D7 is the screen width flag, $D8 is the 40 column text/graphics mode flag, $D9 is the shadow register for the CHAREN bit of $01, $DA-$F9 are used for windowing, screen editing and keyboard reading operations, and $FF is used by the BASIC interpreter. Therefore the only free locations are $FA-$FE inclusive, although you may be able to use some of the others if your machine code routine doesn't jump back to a BASIC program at all. Detailed memory maps can be found in the book "Commodore 128 Reference Guide for Programmers", written by David Heiserman and published by Howard W. Sams and Co, Inc. One reference book that I can recommend is called "Commodore 128 Assembly Language Programming" which devotes whole chapters to assembly language music, and a whole chapter to the 80-column display system. It is written by Mark Andrews, also published by Howard W. Sams and Co., and its ISBN is 0-672-22541-7. Another one is called "500 C128 Questions Answered" and is sold by ESSL in England. The address is Masons Ryde, Defford Road, Pershore, Worcs. WR10 1AZ. I hope that I have been able to provide you with some information that you will find useful. As I say, look out for a suitable article.

RADIO RELATED

Dear CDU,
Could I ask your help on two points? First of all, could you or any reader help with a source of memory expansion for a VIC20. I have just been given the VIC20 which I wish to use as a Log Book leaving it on when in the shack and avoid the need to swap programs on the C64. The only source I have seen was DATEL but they no longer sell them. A used one would do which I am willing to pay for obviously. A suitable program as well would also be very acceptable. Secondly, I have seen references to the fact that the C64 cannot be used in 80-column mode. What puzzles me is, it this is so, how does the Packet program DIGICOM (terrific program) so easily switch between eighty and forty columns? Thanks in advance
Ray Robinson, Co.Durham.

Dear Ray,
Before I answer your query I would like to thank a friend of mine KEN MOTTLEY for asking me a while back to change the program that he was using for a radio log book. Since that time we have written many a letter with regards the updating of the program. It is to him that I owe the fact that I can answer the question knowing the basis of what you are asking. Without that knowledge I wouldn't have been able to give your letter a suitable title either. The log book program that I have written is however for the C64 and if you would like a copy of that then I shall send it to you free of charge when it is completely finished. Also, a bit of free advertising here - if there is any company or organisation that would be interested in distributing a full-feature Radio Log Book program on disk then please get in touch. Now to your queries (at last!). I do not know of a company that still sells the memory expansions for the computer that you mention. However, I would have thought that at least one of our readers will have one that they are prepared to sell to you for a reasonable cost. If that is so then could that person please write to the Techno Info headquarters - address at end. The question of eighty

column mode is one that has arisen on a number of occasions. It is a simple fact that the C64 does NOT have a true eighty column mode. With a bit of clever programming you can use the graphic bitmap screen of the 64 to emulate an eighty column display though. Each character is made up of a grid of four by eight bits. By doing that, two characters can be displayed in each normal one character zone. I must emphasise that it is only a graphical effect and a bit of neat coding that produces eighty columns - it is all governed by the high-resolution bitmap mode.

SORTING SUPERFILE

First may I mention a word of praise to the fourteen year old who is credited with writing the program SUPERFILE in your September 1989 issue. It handles beautifully, as does TEXTED which I am using to write this letter. However I have two problems with SUPERFILE. The first being the sort section which worked when I checked it on a few entries but as I built up the file to around fifty or more entries and asked it to sort them, the program locked up. The other is the printing of labels. It prints the whole label across the page instead of down it. My equipment consists of a C64, a 1541-II disk drive and a STAR LC10C printer. I trust that this info is sufficient for your needs to de-bug the program.
Bill Tisdale, Coventry.

Dear Bill,
Unfortunately the program has been compiled by MINI-BLITZ and so it is virtually impossible to debug. It is not the normal machine code that I know and love or the normal BASIC. It is a code all of its own and I am simply unable to list it or look at it in any way. I will send out a plea to Madhu Surendranath, the author, to send me a copy of the program as it was before he compiled it. If he could send it to me at the Techno Info HQ then I will have a look for you. I am somewhat confused about the label printing though, because my STAR LC10C set-up prints them out fine, vertically down the page. Sorry that I can't be of any help but compiled programs are not particularly easy to look at.

CARTRIDGE WANTED

Dear CDU,
I have a friend that lives abroad and he finds it very difficult to obtain hardware for the C64. He recently visited us in England for a fortnight and he liked my Super Snapshot cartridge, but unfortunately it never dawned on either of us to order him one whilst he was here. As a gift to him I would like to buy a cartridge like Action Replay, The Final Cartridge 3 or Super Snapshot, or something similar that has a machine language monitor and additional functions like new command words and defined function keys. I am not looking for a cartridge that only "backs-up" software. However, because I am on a fixed income I cannot afford to pay the shop prices. Please could you publish my letter in the hope that one of your readers may be thinking of upgrading to another, thus allowing him to sell the one he has already to me at a lower price. I always read your column and think that it is very worthwhile. I hope you can find space for my request.
Peter Appleby, Salisbury.

Dear Peter,
Thanks for your letter. I have indeed published your letter and I would just say that if one of our readers does have a cartridge that they would like to sell then please could that person write to me at the special Techno Info address with any relevant details.

TRACKER BALLS

My first problem is that I bought WordWriter 3 from FSSL. After I print out a document, the tractor feed continues to feed paper and only stops when a blank sheet has been pushed out. This means that I lose a sheet with every document. The printer I am using is a STAR LC10C. My next problem is with a MARCONI Tracker Ball. I bought this second hand with no software and no paperwork. It is the RB2/PC-3. I tried it in both ports using CDU Paint and all I got was a random pointer. Turning the ball left could just as well send the pointer up. I wrote to Marconi and they were unable to help me and referred me to where I bought it. This I was unable to do. A friend has one for the BBC but it appears that my one has an extra board in it so it is not comparable.
T.Eley, London.

Dear Mr.Eley,
I cannot suggest anything for you to do with respect to your first problem as it would mean altering the software. Could you not switch the printer offline when it has finished printing what you want. Then switch the printer off and the program may go back to its editting mode or something. I do not know because I do not have the software. Sorry. The problem with the Tracker Ball is probably not due to incompatibility because if it was, I presume Marconi would have been able to say straight off that it wasn't compatible with Commodores. The problem is that you need to use it with a piece of software that allows the input to be controlled by a tracker ball. The input from them is not the same as a joystick and so CDU Paint would not know what to do. It would be the same effect as trying to use a standard proportional mouse to control a shoot-em-up game. The input received by the computer from the mouse would be so vastly different from what it expects and so

nothing would be done. You need to find a piece of software that is compatible with a tracker ball and unfortunately I don't know of any.

1520 EQUIPMENT

Dear CDU,
Could you please tell me where I can get hold of a lead that will enable me to connect my C128 to my Scart TV. I am not able to get out of the house very often and my searches over the telephone have not been at all fruitful. A few years ago I was given a CBM 1520 printer/plotter but the pens have now run out, as has my supply of paper. It seems a shame to have this nifty little piece of equipment lying dormant. Can you also please tell me from where I can obtain pens and paper for the plotter because it is old and my local computer shop doesn't seem to have heard of it! I would be most grateful of any information that you may be able to supply.
Wayne Jones, Nottingham.

Dear Wayne,
There is a company based in Merseyside that I can thoroughly recommend to you for all three pieces of equipment. I have found that they are one of the best sources of general hardware and what's more, their products are competitively priced which is an added bonus. They are called Meedmore Ltd, the address being 28 Farriers Way, Netherton, Merseyside, L30 4XL. Their telephone number is 051-521-2202 and they stock a suitable lead and the pens and paper for the plotter in question. They also stock a huge range of printer ribbons and other leads. They have always been very helpful in the past and I would suggest that you telephone them to check up on the order numbers before you write.

1570 OR 1571?

Dear CDU,
I read Techno Info with interest and hope that you or your readers can solve a small problem for me. Some time ago I bought a second-hand Commodore 1570 disk drive which has seen much service with both C64s and my C128. With it came a 1571 users guide and the advice that "it's practically the same so this will do". At the price I paid I'm not complaining, it's just that it very obviously isn't the same as a 1571. I have found references all through my literature to the 1541, 1551, 1581, 2031 and 4040 disk drives but never a mention of the 1570. If you or any of your readers could satisfy my curiosity, or provide me with a copy of the users guide it would be appreciated. Thanks, and since I discovered your magazine last November it features firmly on my monthly shopping list.
S.T.Curtis, Bristol.

Dear Mr Curtis,
I am glad that you have discovered the best serious magazine for 64/128 owners and I hope you continue to enjoy it. But I must disappoint you - the 1570 and the 1571 ARE practically the same. (That's a sign for people to provide lists of striking differences!). The User's Guide, at least, is the same for both peripherals usually. My 1571's manual has "DISK DRIVE - 1570/71 User's Guide" clearly stated on the front. Perhaps your manual is different. The drive certainly isn't similar to the 1581, 2031 or 4040 and so references to other drives don't matter. I would have liked you to have told me what it was about the drive that made you say they were totally different. I must confess that I am not 100% familiar with the 1570, and that I am just going by the manual that I own and my other reference books. Sorry that I cannot satisfy your curiosity completely.

NIBBLER NOBBLED

Dear CDU,
Summat odd, I thought. I rang DATEL ELECTRONICS with a query about my Burst Nibbler, and was told (in passing) that sale of this utility was now "not allowed". I know of no legislation that could be invoked to ban the product that would not apply equally to photocopiers or cassette recorders. Can you tell me who imposed such a ban and under what law it is enforced? Incidentally, I see that you are recommending GEOS to Jim Morris who was looking for "faster everything". GEOS is fun, but I'd never think of using it as a word-processor. Too much preparation involved. Try MINI OFFICE II!
Brian Caukwell, Bancroft.

Dear Brian,
I put out a few feelers for you and also talked to a pleasant chap from DATEL. It would appear that your nibbler has been banned, quite simply, under the 1988 Copyright Act. It is illegal to manufacture, sell or even use anything whose primary or sole purpose is that of copying commercial software, or so it appears. The Burst Nibbler has no other function than to copy commercial software. Photocopiers are meant to be used perhaps for copying reports and documents that you have produced yourself, such as Curriculum Vitae. Cassette recorders can be used to play cassettes. That is why things like Action Replay, DATEL's excellent cartridge, haven't been banned. They have other useful functions outside that of copying. I am not a solicitor and so that is all I can say. I don't wish to be drawn into a great legal debate either. But if any solicitors are reading this and know the exact grounds then, please, do tell. I recommended GEOS to Jim Morris because he wanted something that also had icons - Mini Office II doesn't really have that many.

NEW IDEAS

Dear CDU,
I am seriously considering attaching an Amiga disk drive to the Commodore 64 and writing a full DOS for it, and this is where you come in. I intend to attach to the Expansion Port the following chips: A 6526 CIA located at $DF00 to $DF0F, A 6526 CIA located at $DF00 to $DF0F and an 8K ROM chip located at $8000 to $9FFF. If you can supply a schematic diagram for this circuit, or can give me the address of someone who can, then I will tell everyone how wonderful you are. Many thanks for accepting my past queries.
Mark Carroll, Cornwall.

Dear Mark,
An offer I can't refuse! I personally can't supply a schematic diagram because I was not blessed with such luxuries. I don't know if a Commodore User Group would have one, but you could try ICPUG on 081-346-0050 to see if they can supply one. Failing that, a plea! Could a very kind person please send us a diagram of the sort that Mark is after, and I will pass it on to him. I wish you the best of luck with your project

FAST LOADERS

Dear CDU,
I am a 15 year old boy and I do not know too much about computers. Recently I bought a 1541 disk drive and was shocked at how long software took to load. But I have heard about a 'fast loader' - Dolphin DOS. I am not too sure what this does, please could you explain certain things. For example, how long does it take to load software, do they fit in the disk drive, do you experience problems loading certain software, where do you buy them from, can you fit them yourself or does you drive have to be sent off and how much does it cost??? I would be very grateful for any information supplied.
Scott Smith, Doncaster.

Dear Scott,
There would be no point in my telling you all about Dolphin DOS because it is no longer widely available and you may have trouble finding it. A similar sort of thing that is available today is called "JiffyDOS" and it can be purchased from FSSL (address under '128 SPECIFICS'). It speeds disk access up by a factor of around fifteen times. It is supplied as two chips, one to replace the Kernal ROM in the computer, and the other to replace the DOS ROM in the disk drive. It is a simple matter of replacing the two necessary chips with the ones provided. You even get some improved disk handling commands into the bargain. It is compatible with most other hardware and software and you fit it yourself, no need to part with your drive. The product for your computer and one disk drive costs £59.95 but don't gasp, please! The Dolphin DOS was expensive as well.

STARTING OUT

Dear CDU,
I am interested in learning how to program so that I can make my own games. I have a C64 Light Fantastic package. Any help in understanding machine code would be most helpful and any other information on how to program if you can please.
Rick Salmon, Manchester.

Dear Rick,
This is really a difficult thing for me to do. I cannot explain in such a small space everything you need to know about programming. You don't mentioned how well you grasp BASIC at the moment. Before you even attempt to program in machine code you should make sure that you are fully conversent with BASIC. Some people may not recommend the same to you but I would. Also get as many books as possible on the subject - the Programmer's Reference Guide is a good start together with things along the lines of "Introducing BASIC". Your local library should have plenty of books on the subject. The key to learning how to program is experimentation. And the key to learning how to program in machine code is to look at how other people do simple things and experiment with their routines. Examine them and try to work out what each part does. I'm afraid that I can't really provide any pinpointed advice on programming. It is a very general topic. I just suggest that you keep reading CDU and look out for the serious features on programming. One called "Exploring BASIC" was featured a little while back. You may find that useful.

MICRO-WHO?

Dear CDU,
I hope you can help me. I have been having a few problems with a Microline micro-83A printer. Could you please tell me what Microline's address is so that I may contact them.
Mark Chilvers, Great Yarmouth.

Dear Mark,
Searching through books and asking around the CDU office has proved totally futile - no-one seems to have heard of Microline unfortunately! There seems to be Micro-everything apart from Micro-line. Not even that great emperor, Paul Eves, has heard of them. I can only ask that if a reader knows where they are based now, if they are still in operation, could they please forward the address to us.

FORMATTING PROBLEMS

Being very new to computers, my knowledge is very limited. But I am enjoying the challenge of understanding how computers work. I have a C64 with 1541 disk drive. I used your fast formatter from volume 1, issue 2 to format a few disks. On the disk with the formatter was Menu Maker. The fast formatter I can do without but the menu maker I found very useful. However, while formatting I forgot to change disks over and formatted the disk in question. Could you tell me where I could get hold of a copy of Menu Maker or advise me on programming my own if that is possible for a novice! I would be very grateful if you could help me.
G.A.Tampling, Oxfordshire.

Dear Mr.Tampling,
I can't really provide any advice on programming your own menu maker because it is a very complex and involved task. Instead I shall send you through the post a copy of the disk that you formatted by mistake. How's that? Incidentally, everyone has done that in their computing career. I had a phase where I used my 1571 to reformat disks. My brain just wouldn't accept the fact that it formats both sides of the disk in 1571 mode. Consequently I lost the flip side of about three disks.

UPDATE

Many more people wrote to us with details of PASCAL packages for NEIL MARSHALL. I can't mention all your names - you know if you sent information. But I must thank GEORGE HIGGINS OF STEVENAGE specially. I was overcome by his amazing generosity, supplying Neil with an original copy of SUPER PASCAL with disks, manual and hard-backed case absolutely free of charge. Thank you very much. It is always very nice when our readers are so kind.

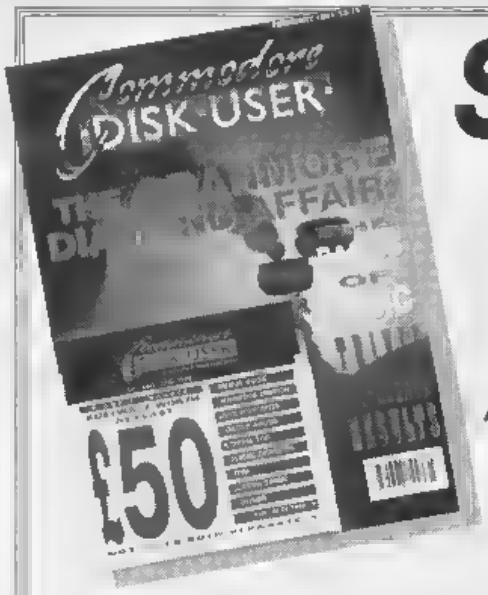
Thanks also to STEVE LINES OF CHELTENHAM who wrote regarding the problem from GERRY SMITH OF CHESHIRE in January's Techno Info. He experienced the same problem and it was due to a malfunction of one of the chips. It was IC U2 which is the 6526 on the left of the two, he says. Indeed it is. The "U2" is the reference code that you will find printed on the circuit board next to the chip (one of the big ones!).

TIP OF THE MONTH

This month's tip is quite short and comes to you from DAVID MACDONALD OF PETERBOROUGH and concerns the SMOOTH SCROLL DEMO program from issue one of this volume:

For readers puzzled as to altering the character in the moving background demo as I was, I suggest they alter line 29040 to read: 29040 POKE 40001,N: POKE 40003,N where 'N' is the character number they wish to use. Also, in lines 20120, 20140 and 20395, if the spaces in between the words are deleted and replaced with cursor right control codes, the moving background moves between the words and improves the appearance slightly.

Thanks for that David. Unfortunately, that is all we have time and space for this month. If you have any general queries or have been experienced problems of any sort related to hardware or software then drop us a line at CDU TECHNO INFO, 11 COOK CLOSE, BROWNSOVER, RUGBY, WARWICKSHIRE, CV21 1NG. That is also the address to which you should send your tips for Tip of the Month and your entries for the Techno Info challenge detailed last month. Next month there will be some news about "Full Disk Jacket", the program by Mike Gregory that had a number of people asking about how to make it compatible with a STAR LC10C printer. See yer then!



Subscribe now . . . And Save £8

OR KICK YOURSELF FOR THE REST OF THE YEAR . . .

We've gone mad and are offering you a once only opportunity of receiving Commodore Disk User's next twelve issues for the staggeringly low price of *£25 if you live in the U.K. We will even post it to you free as well.**

Published monthly –

COMMODORE DISK USER is the answer to every Commodore computer owner's dream. The disk supplied with the magazine contains a variety of ready to use, high quality computer programs – no more lengthy typing in of listings. The scope of the programs is wide, varying from games to business software and high-powered disk utilities – and the disk would retail for at least £50.00 if bought independently.

Of course, that isn't all. The magazine, besides containing full and comprehensive instructions for using the disk, is a complete computer journal in its own right, with news, reviews, programming, competitions and general interest features.

Don't Delay Respond Today

PRIORITY ORDER FORM

Please commence my subscription to Commodore Disk User with the ..... issue.

I enclose a cheque/postal order for £..... made payable to **ALPHAVITE PUBLICATIONS LTD.**

or debit £..... from my Access/Visa Card No:

Valid from ..... to .....

Signature ..... Name .....

Address .....

.....

.....Post code .....

Cut out and send this form with your remittance to.

Subscriptions Manager, Alphavite Publications Ltd., 20 Potters Lane, Kiln Farm, Milton Keynes, MK11 3HF.

* Rates refer to subscriptions sent post free to UK addresses. Overseas rates on request

COLOUR TABLE EDITOR

Take the hassle out of producing colour tables SIMON COLLIS

In the DECEMBER 1990 issue of CDU we gave you COLOUR TABLE EDITOR. Unfortunately, we managed to put the wrong program on the disk. We did in fact give you an incomplete version of the program. My apologies to everybody, and to SIMON COLLIS in particular.

PLEASE NOTE that the demonstration files on the DECEMBER issue of CDU are NOT compatible with the program given here. We provide a new set of demos on this months disk for your enjoyment.

ABOUT THE EDITOR

The colour table editor was written for a specific purpose. I wanted to design a huge colour table for an intro sequence. Then I wondered whether the readers of CDU would be in the same position, not wanting to reassemble the same program hundreds of times merely to test a colour table

I therefore set about writing this program, bearing in mind that people may want varying widths of raster line, and varying sizes of table.

LOAD AND GO

The finished program allows the width of the raster bar, the BAR SIZE, to be altered easily, keys B and Shifted B. The length of the table can be altered as well using the keys T and Shifted T. The table will also scroll upwards when requested. This feature can be turned on and off using the S key. During editing, the table will scroll to the position in the table that the cursor currently resides at. However, some may find this a hindrance. In this case locate the cursor in a place in the table you wish to view and press L. Once you do this, the table locks in that position until you press L again.

To change any value in the table, you simply use the + and - (plus and minus) keys. Note that Shifted + is the same as - and Shifted - is the same as +. This is due to the way that these keys are programmed, and is also useful for people that only want to use one key. (like me!)

DIFFERENT MODES

There are two modes for viewing the full or edited table at the top of the screen. This first is the default mode, NUM, and displays the colours as numbers. The second is COL and displays the edit table as blocks of colour. In the NUM mode, the cursor is white whilst in the COL mode it is a star. You can alternate between these two modes using the V key.

Should you wish to clear out all of the table, simply change the current value in the table to the colour the table to and press CTRL C. CTRL T is the final option I should mention. The value under the cursor becomes that last value accessed in the table when CTRL T is pressed.

Key H takes you to the HELP page and F1 takes you into the files mode. From the file mode, you can access drive 8 to store tables, display a directory, use DOS commands etc. All the keys available are displayed on screen.

Explaining all about COLOUR TABLE EDITOR could take the whole magazine, simply TRY DIFFERENT THINGS and see what happens.

You can load COLOUR TABLE EDITOR simply by typing LOAD"colour table editor",8 and typing RUN or select it from the CDU menu.

6510+ ASSEMBLER

DAVE WEAVER's Assembler gets another airing to compliment the new series on Machine Code programming - brought to you by JOHN SIMPSON

The following assembler instructions where originally published in CDU in the MAY/JUNE 1989 issue of CDU. We are reproducing them now so that any new readers to the magazine can benefit from DAVE's assembler, also, it will compliment the series for newcomers to the world of MACHINE CODE programming. On the original 6510+ assembler, the function for printing to a printer had a bug and would not function correctly. The program on this months disk has this bug removed and the print function now works. Thanks DAVE.

ABOUT 6510+

This assembler is a valuable aid both for writing professional machine code programs and for learning about programming. It is a three-pass assembler which allows the use of labels and contains extra commands that speed the production of code by permitting merging routines from tape or disk, finding and changing given strings, deleting of redundant lines, auto line numbering and, as you will see, a host of other commands. Once the code is assembled, an in-built memory monitor can be used to save or modify the raw code.

Before looking at this powerful programming tool, we'd like to say thank you to COMPUNET for making this program available and a special thank you to Dave Weaver for writing such a beautifully logical assembler.

The 6510+ is a powerful three-pass, disk-based assembler/editor for the Commodore 64

It features:

Standard 6502 mnemonics and addressing modes.

An advanced Pet-like, machine code monitor built in.

Enhanced screen editor, including FIND, CHANGE, MERGE and many more commands.

User definable function keys.

Assembly from disk

Source-code compatability with Supersoft's popular MIKRO assembler.

LABELS

A label is an alphanumeric string of uppercase characters, the first of which must be a letter (A-Z). It can be any length (well, up to 250 characters, theoretically, but it is physically impossible to enter a label of much more than 70 characters on a line of source code).

COMMENTS

A comment can appear either on a line of its own, or on the end of another line. The comment must start with either a semi colon (;) or an exclamation mark (!).

Any text entered after a comment is not tokenised by the Basic interpreter. This has the unfortunate side effect that any PRINT commands used whilst using 6510+ will report errors if they contain a semi colon. This shouldn't cause too much of a problem. After all, who uses Basic?

ASSEMBLER DIRECTIVES

In addition to the standard 56 mnemonics, the assembler accepts certain other three-character commands during assembly, namely BYT, TXT, WOR, END, OUT, OFF, CHN, LNK and LIB. These operate as follows:

BYT is used to reserve one byte o memory and load it with a value. BYT directives may contain a series of comma-separated byte values, which will be stored in consecutive memory locations. ASCII strings may be generated by enclosing the string in double quotes.

```
BYT 2, 3 FRED
BYT 'HELLO WORLD!'
BYT 5+4, 'YES', 0
```

All values must be single byte values, they must therefore be between 0 and 255.

TXT is included from MIKRO compatability. It is equivalent to the BYT instruction.

WOR is used to reserve and initialise two bytes of data at a time. Each value in a WOR command is considered to be a two-byte value (0-65535) and is stored in standard low-byte-first format.

```
WOR $1234
WOR %1100101011001
```

The first example would be stored as two bytes: $34 and $12.
END indicates the last line of source code. Any lines after an END directive will be ignored by the assembler during assembly. This is optional if it is the last line of the source code.
OUT causes a listing to be generated on the third pass of an assembly from the line of the OUT command onwards. The listing is produced on the screen but if you would like a listing on a printer, enter OPEN4, 4:CMD4 before assembling the program. This redirects the screen output to the printer. Please note that this is not exactly the same as MIKRO's OUT command.
OFF turns off a listing (started with OUT) for the rest of the asssembly, or until another OUT command is found.
CHN and LNK are equivalent commands that allow several source files to be 'chained' or 'linked' together. This command terminates assembly of the current file, and loads in the specified file. There are no restriction on the number of files that may be chained in this way. The last file in the chain must use an END command followed immediately by the name of the first file in the chain. In this way the next pass can begin with the correct file!

```
file 'PART1': 10 INC ERED
              20 RTS
              30 CHN 'PART2'
file 'PART2': 10 ERED=53280
              60 END 'PART1'
```

LIB allows you to insert source code from another file into the assembly. When the assembler encounters the LIB directive, it temporarily stops reading source code from memory, and reads a line at a time from the file named. Processing of the in-memory, source resumes after either an end of file or an END command is encountered in the LIB file.

```
file 'ONE' 10*=49152
           20 ERED=53280
           40 END
           30 LIB 'TWO'
file 'TWO' 25 INC ERED
           99 RTS
           100 END
```

This command allows you to make your code much more modular. In fact the 'main' program could consist of only a series of LIB calls.

```
1000 *=49152
1010;
1020 LIB 'START'
1030 LIB 'MIDDLE BIT'
1040 LIB 'SOMETHING ELSE'
1050 LIB 'THE END'
1060;
1070 END
```

EXPRESSIONS

An expression can be used at almost any point that a single number could be used. It consists of one or more numbers/lables, each separated by one of a group of mathematical operators as shown in Table 1.

TABLE 1

OP	PURPOSE	EXAMPLE	RESULT
+	Addition	10+4	14
-	Subtraction	$1a-11	15
*	Multiplication	%1010*13	130
/	Division	54/10	5
%	Mod (remainder)	54%10	4
&	Bitwise AND	6&3	2
^	Bitwise OR	6^3	7
>	Bit shift right	1>4	%10000
<	Bit shift left	%10110<2	%101

The following unary operators are also provided:

OP	PURPOSE	EXAMPLE	RESULT
'	take ASCII value	'A	65
<	take low byte	<$1234	$34
>	take high byte	>$1234	$12

All operators have equal precedence.

A $ is used to indicate a hex number, and % is used to indicate a binary number. A number with neither a $ or a % is assumed to be decimal.
All expressions are evaluated in left to right order. Brackets may be used in an expression to force the order of evaluation to be other than left to right.

```
1+2*3=9
1+(2*3)=7
```

The fact that three of the operators (%, < and >) are used for two different things may appear confusing at first, but it is quite apparent which action is meant from the context in which the expressions appear.
Two special characters (*and @) may also appear in expressions. These have the values of the PROGRAM COUNTER and the AT COUNTER respectively. These will be explained in more detail later.

```
FRED = $1234+4          $1234
LDA < FRED+2              $36
BLAH = $100*(2+3)        $500
XXX = 50/10                5
LDY # 3< XX      %1100000(96)
```

THE PROGRAM COUNTER

In order to tell 6510+ which area of memory you wish to assemble your code to you need to set the

PROGRAM COUNTER (the * variable) to the address required.
For example, to assemble your code so that it is placed to run at address 49152 onwards:

```
10 *=49152
20 ...rest of code
```

During assembly the *variable will always hold the address for which the current instruction is being assembled. This enables you to program simple branches without the need for labels.

```
240 CMP # 10
250 BNE FRED
260 INY
270 FRED STY SOMEWHERE
```

could be written as:

```
240 CMP # 10
250 BNE*+3
260 INY
270 STY SOMEWHERE
```

Because in the first example, FRED will always be three bytes further on than the BNE instruction.
Now, consider the following problem. You have written a program (such as an amazing assembler to rival 6510+) which needs to be assembled at address %8000 onwards.
If you put a *=$8000 in your code, it would be assembled to this address but this would put it in the same area of memory as 6510+ which would then be overwritten (although 6510+ will recognise this fact and warn you).
The solution is to use @, the AT-COUNTER. This is similar in concept to the program counter but, whilst the program counter tells 6510+ the address at which the code is to run, the AT-COUNTER tells 6510+ where in memory to place the final assembled version.
One answer to the above problem is to use:

```
10 *=$8000
20 @=$4000
30 ... rest of code
```

This would cause 6510+ to assemble the program as if it were to run at $8000, but the final assembled code will be placed in memory at $4000 onwards. The program can then be saved to disk using the monitor, the computer then switched off and on (to remove 6510+) and the program loaded in and moved to $8000 where it can finally be run. (A bit long-winded I know, but it works).
There is an alternative way to set up the AT-COUNTER, which is included for MtKRO compatability. This previous example can also be written as:

```
10 *=$8000, $4000
20 ... rest of code
```

Note that setting the program counter will also set the AT-COUNTER to the same value. So, if your're using the AT-COUNTER (you won't normally need to) then remember to set up @ AFTER setting up *

EDITOR ENHANCEMENTS

Anumber of additions have been made to the way the normal screen editor works while using 6510+.
The left SHIFT key may be used to pause output to the screen. For instance, when listing the source code, the SHIFT LOCK key may be used as a pause and hold key.
When the RUN/STOP key is pressed the quotes mode and number of outstanding inserts flags are set to zero.
SHIFT + will put the cursor in the bottom left corner of the screen, like a sort of un-home key.
A DOS wedge routine has also been included. Entering @ will give the disk drive status. Typing @ command will send 'command' to the disk drive. Typing $ will display the disk directory, without actually loading it into memory. The $ can also be followed by a wild card to give a partial directory. The default device is used (see later).
For example, to format a disk type:
@N:NEW DISK, OK
to display the disk directory:
$
to display a directory of all sequential files beginning with the letter A:
$0:A*=S
=S gives just SEQ files and A* gives files beginning with A
6510+ also allows the eight function keys to be defined to hold any string of up to 31 characters. More of this later.

BASIC EXTENSIONS

6510+ adds over 25 new commands to the existing Basic ones.
With 6510+, any Basic commands will now accept hex and binary numbers, as well as decimal numbers, by preceding them with a $ and a % respectively. So the following are all valid, using 6510+:

```
PRINT $123*%1010
PRINT CHR$ ($40)
```

Now onto the new commands. In this section any item in square brackets is optional and may be left out. All commands may be abbreviated as in Basic (A shift-S instead of ASSEMBLE).

EDITOR COMMANDS

OLD
This is the opposite of NEW. A program that has

been NEWed can be recovered using OLD.

AUTO (line-number [,step])
AUTO will present line numbers automatically when a program is entered. The number presented will be the number of the previous line plus the current step value. Auto line presentation is turned off by pressing return on a blank line. If no step is given the value of 10 is used. If no start line is given the value 1000 is used.

RENUMBER (start-line [,step])
This will renumber a program starting at the given line number, each time adding the given step to produce the next line number.

DELETE line-range
DELETE will remove sections of the current program. The line-range given is in the same format as the Basic LIST command.

```
DELETE 1230-2000
DELETE 100
DELETE -1294
```

FIND XstringX
This command will search the source code for the string given. Any lines containing the string will be listed to the screen. X is any character not included in the string.

```
FIND HELLO
FIND/LDA/
```

CHANGE XstringX replacementX
This will search the source for the given string and replace it with the replacement string. Each line where a change is made is listed to the screen.
CHANGE @ HELLO@HELLO WORLD!@
Changes all occurrences of HELLO to HELLO WORLD!.
CHANGE"!""
Remove all exclamation marks from the source.
It is important to remember that the exclamation mark (!) and semi colon (;) are used to start a comment in 6510+ source code, so any characters following these will not be tokenised. This can cause some problems with the FIND and CHANGE commands. For example:
CHANGE /!/*/ will NOT change all exclamation marks to asterisks. This is because the /has two different values in the line above. The first is tokenised into the divide token. The next two are not tokenised since they follow an exclamation mark. Instead use CHANGE"!"*" This will work since the exclamation mark is not taken as the start of a comment starter, because it is in quotes, and everything in quotes is taken literally.

FUNCTION KEYS

KEY
This will display the strings currently attached to the eight function keys. A <left-arrow> in the string represents a RETURN.

KEY number, string
This form of the same command will let you change the key definition to anything you choose. Only the first 31 characters of the string are used.
KEY 1, "old"<left-arrow>renumber<left-arrow>
(The <left-arrow> is used to insert RETURNs in the string)

KEYSAVE "name"[,device]
This will save the current key definitions to disk or tape.

KEYLOAD "name"[,device]
This will load a key definition file from disk and re-program the E-keys accordingly. The default device number is used if none is specified.

KEYOEF and KEYON
These commands will disable and enable (respectively) the new function key routines.
This is useful for those lucky people who have alternative operating system ROMs installed (such as those supplied with parallel DOS systems) which have their own F-key definitions.
With Trilogic's PHANTOM parallel DOS (which is all I've tried 6510+ with so far), if the key routines are enabled (KEYON) and a key is defined as nothing (KEY1, ") then the default PHANTOM definition is used instead.

HELP
This command will display a list of all new and modified commands.
It is only meant as a brief reminder.
For more details read this documentation carefully.

DISK RELATED COMMANDS

LOAD "name"
SAVE "name"
VERIEY "name"
These commands have been modified sothat the default device is used (usually device 8 - the disk drive). See the DEVICE command later on for more details.

TYPE "name"[,device]
This will read the given file and display its contents on screen. TYPE will only work with SEQ files. The default device is used i none is specified.

DUMP "name"[,device]
This will display the named file in hex and ASCII. DUMP will work with PRG, SEQ and USR files. The default device is used if none is specified.

MERGE "name"[,device]
MERGE will read the named file, one at a time, and enter each of the lines as though they had been typed at the keyboard. In other words, the named file will be MERGEd with the current program in memory. If the same line number exists both in the file and in memory, the one from the file will over-write the one in memory.
Once again, the default device will be used if no other is specified.

APPEND "name"[,device]
This command is very similar to the MERGE command but the named file is APPENDed (added to the end of) the one in memory. Line numbers from the file are not changed so it is advisable to RENUMBER your program after using APPEND.

DEVICE [device]
This command sets up the default device number which is used by all of the disk-based commands in 6510+. If the device number is not specified then the current device number is shown.

ASSEMBLER COMMANDS

These are what 6510+ is all about. In this section EXPRESSION means a mathematical expression. It may contain labels, numbers and operators. Some valid expressions:

```
10
ERED
$1A+(LINE*40)
%1010+>SCREEN
```

ASSEMBLE [line number]
This will assemble the source code currently in memory. If a line number is given the assembly will start at that line, otherwise it will start at the first line of source. Assembly can be stopped at any time by pressing the RUN/STOP key.

DISASSEMBLE <expression>
This will display a disassembly of memory from the address specified in the expression. Disassembly is stopped by pressing RUN/STOP and the left SHIFT key or SHIFT LOCK can be used to pause the listing.
DISASSEMBLE may be abbreviated as D shift-I.

```
DISASSEMBLE START
DISASSEMBLE 4096*12
```

NUMBER <expression>
This will evaluate the expression and display the result in hex, decimal and binary. It is useful for displaying the value of a label or for converting between number bases.

TABLE
This will display the symbol table, from the last assembly, in alphabetical order. Each label is followed by its hex value

SYMSAVE "name"[,device]
This will save the symbol table to disk. There is not much use for this yet but it is included in case I decide to write some accompanying utilities, such as a symbolic debugger, which would need the symbol table.

FORMAT <line range>
This command is very much like the LIST command except that the listing is neatly formatted. Try it and see.

SET <label> = <expression>
This command allows you to manually add to or modify symbols in the symbol table.

```
SET BANANA=FRED*2
SETX = $2345
SET LO= < ADDRESS
SET HI= > ADDRESS
```

MODIFIED COMMANDS

Some existing Basic commands have been modified for use in 6510+

```
POKE <expression>, <expression>
PEEK (<expression>)
SYS <expression>
```

These commands now use the expression evaluator built into 6510+. This means that hex numbers and labels can now be used.

```
SYS START
SYS GO+3
SYS 4096*12
PRINT PEEK (COUNTER)
POKE $D020,0
POKE FRED,<VEC: POKE ERED+1,>VEC
```

SAVE ("name"[,device])
The SAVE command has been modified to provide a useful autonaming facility
When provided with a name and device number, SAVE works as usual and uses the default device number if none is specified. If no name is given, the first program line in memory is examined. If it begins with a comment symbol (exclamation, semi colon or REM) and the next character is a double quote, then the file name is taken from there.
This means that each of your programs can contain its name in the first line, and you don't have to worry about remembering what it was

```
10;"@:PART1",8
1!"@"TEST"
5 REM "@:HELLO",8
```

Notice that the names include '@:'. This is so that when you type SAVE the program will replace the current version on the disk.

LOAD ("name"[,device])
VERIFY ("name"[,device])
These commands have been modified so that they use the default device number set up by the DEVICE command. If no name is specified '*' is used and the first program on the disk directory will be used.

IMPORTANT NOTE

Because of the way these commands are modified, you may find that running ordinary Basic programs within 6510+ isn't necessarily a good idea. This is because the POKE command (for Instance) no longer uses the Basic expression evaluator and no longer recognises Basic variables
The following program would not work using 6510+:

```
10 FOR 1=0 TO 255
20 POKE 1024+1,1
30 NEXT
```

You would get an ?UNDEFINED LABEL error in line 20. But you could use:

```
10 SET X=0: EOR 1=0 TO 255
20 POKE 1024+X,X
30 SET X=X+1
40 NEXT
```

THE MONITOR

6510+ contains a built-in machine code monitor. To enter the monitor type: **MONITOR**
The monitor will then display the current register values, and present you with a full-stop as a prompt. All monitor commands are a single character, usually followed by some hex parameters. In this section <addr> contains up to four digits representing a memory address in hex

D<addr> (<addr>)
This will disassemble the memory between the two addresses. If the second address is not given then only one line o disassembly is shown.

F<addr> <addr> <value>
This will fill the memory between the two addresses with value, where value is a number in the range zero to FF.

T<addr> <addr> <addr>
This will transfer the block of memory between the first two addresses to the area beginning at the third address.

H<addr> <addr> <value> (<value>...)
H<addr> <addr> 'text
Hunts between the addresses specified for the series of values given.
In the second form, a text string may be given if preceded by an apostrophe The monitor will search for the text supplied.

```
H 1000 2000 A9 00
H 1000 2000 'HELLO
```

M<addr> (<addr>)
Displays the memory range given in both hex and ASCII.
To modify the memory contents, simply move the cursor over the hex number to change, type the new value and press RETURN.

R
Displays the current register contents, in the form shown in Fig.1. Any of the values may be changed simply by moving the cursor over the current value, typing the new value and pressing RETURN.

```
Prog  IRQ    Status  Acc   X    Y    Stack
Cntr  vector   reg          reg  reg   ptr


PC    IRQ    SR   AC  XR  YR  SP
;1234 EA31   30   10  A1  00  E5
```

G(<addr>)
This command, Go, will execute the machine code routine starting at the given address. If no address is given, the value in the Program Counter (PC) will be used.

L"name"[,device]
V"name"[,device]
S"name",device,addr1,addr2
These commands will Load, Verify or Save blocks of memory. The L and V commands will use the default device if none is specified.
The S command saves the area of memory between ADDR1 and ADDR2-1. Always remember that addr2 must be the address immediately after the last byte to be saved.
When a program is assembled, the start and end addresses of the assembled code are displayed liked this:

```
START ADDRESS: $2000
END ADDRESS: $2134
```

To save the assembled, executable code, enter the monitor (with the MONITOR command) and type:

```
S"PROGRAM",08,2000,2134
```

X
This command exits the monitor and returns to the assembler.

FINALLY

That seems to be it! I hope I've not left anything out but if you do find something I've not mentioned or something you want explaining, or even, perish the thought, a bug, then feel free to contact me via COMMODORE DISK USER. Even better, if you're on Compunet send me an MBX (my ID is DW28).
This might be an opportune moment to mention the fact that all design and programming was done by me (Dave Weaver), with inspiration taken from Supersoft's MIKRO assembler. I hope you enjoy using 6510+. THE REMAINDER OF THIS ARTICLE, NAMELY THE ERROR MESSAGES POSSIBLE, WILL BE PRINTED NEXT MONTH

BASIC MACHINE LANGUAGE TECHNIQUES

A new series gets underway for Machine Code programmers

JOHN SIMPSON

In the words of John, we bring you "An Introduction to a series of lessons designed for the beginner to enter the world of machine code programming on the commodore 64."

BUILDING WALLS

If you want to build a brick wall across the bottom of your garden and you go to a builder and say, "build me a brick wall X feet long by Y feet tall and consisting of red brick", then this can be likened to the BASIC language. On the other hand, if you calculate how many bricks you required, plus sand and cement, go to a builder's merchant to purchase the materials needed and then proceed to build the wall yourself, this is analogous of Machine Language (ML). This is not, I must admit, the best of analogies but it does tend to show the difference between a High Level Language, such as Basic, and the Low Level of Machine Language. In the first you don't really require any bricklaying knowledge and the result is a wall of pretty much a standard type - plus you may have to wait quite a while before the builder can 'fit you in' (no slur on builders intended!). On the other hand to build the wall yourself will require a good understanding of working with bricks, assessing and costing materials, as well as design. The end result can be a more custom-built and designed wall, probably constructed more quickly and aesthetically much more pleasing. However, a good deal more effort is needed.

OPENING DOORS

To work proficiently in machine language and to produce the individuality and high-speed smoothness of a typical program requires a good knowledge of the machine itself; the bus address lines and data lines, the ALU, the microprocessor, flags and the stack, memory mapping and etc. Actually programming each instruction, however, is rather simple, like picking up a brick, however, it does require a lot of brick-picking.

For example Basic might be - go to the door, open it, walk through the now empty space, then close the door.

On the other hand Machine language would be - place leg One in front of leg Two, shift balance, swing leg Two forward, shift balance, swing leg One forward, repeat this 'n' times, lift right arm to 'n' height, open hand, grasp door-knob, turn door-knob right 45 degrees ... and so on.

The end result is absolute control over the machine, and, therefore, over the end user of your program. What this dictates is much less chance of 'user-interference', random 'bugs', and system failures during run-time.

FROM INTERPRETERS TO HIGH SPEED COMMUNICATION

Basic is rather like going to a land where the language is alien to you and then attempting to conduct a highly detailed, in-depth, conversation with a person of that land who cannot understand your language! To converse with each other you must employ the services of an interpreter. First you speak - the interpreter translates to the other person. This person absorbs and in turn speaks - the interpreter translates to you... Time consumingly slow!

Machine language, on the other hand, is the equivalent of two people who understand each other perfectly and without ambiguity conversing in 'rapidSpeak'. Very fast and very efficient.

THE LANGUAGE OF THE MACHINE

The actual language of the machine is carried out entirely by the use of small electrical switches which are either open or closed, on or off. These switches are represented by using the binary language (this was discussed in the October 1990 issue of CDU - Numbers and Bytes). Here 1 represents On and 0 represents Off. Let's take a look at an example:

BASIC POKE 1024,2

MACHINE CODE
10101001000000010100111010000000000000100

Gulp! Even a small routine would be so vast and complicated we might as well switch off the 'puter and go down to the pub for a pint (unless you're under-age, in which case it's a milky shake for you!).

ASSEMBLERS AND MNEMONICS

There's always a better way, and this is where an ASSEMBLER leaps to the forefront as the ML Programmer's trusty trowel for laying down bricks of code. Let's break down the piece of code described above into separate eight bit chunks and examine each of them more closely.

10101001 This is represented by an Assembler instruction code also called a mnemonic - LDA which means LoaD the Accumulator with the value of the next byte. (Don't worry about what the mnemonics mean or do just yet, we'll be discussing these later, once the lessons truly begin).

00000010 The value of 2. This value is placed into the Acumulator.

10011101 Another Assembler instruction code or mnemonic - STA which means STore the value held in the Accumulator into the sixteen-bit memory address to be held in the next two consecutive bytes.

00000000 Zero, this is the low byte value of the sixteen bit address.

00000100 Four, and this is the high byte value of that address.

There are 56 assembler Instruction codes and I have listed these in TABLE 1. (you will probably refer to this table quite often as you develop your programming ability). However, now, and using the same piece of code once more, it can be written using Assembler Code - commonly referred to as Source Code. Thus:

```
LDA #2
STA 1024
```

Once you know what the mnemonics (LDA and STA) represent then the above is certainly a lot more understandable and easier to use than the original machine code, i.e.

1010100100000010100111010000000000000100

Once you have written a routine or program and have entered all of the Assembler Code, and everything is to your satisfaction, you can then begin to 'assemble' it. The Assembler will check through all the mnemonic instructions, labels, definitions, and etc., and convert them directly into machine language - which is referred to as Object Code. The Object Code can then be saved out as a binary machine language program.

A CLOSER LOOK AT ASSEMBLERS

To program with ease and efficiency in machine language an Assembler is necessary (it can be assembled by hand but believe me you won't want to). If you already own an Assembler, fine, but more probably, if this is your first excursion into machine language programming, then you might not. Therefore, in all his wisdom, our illustrious editor, Paul Eves, has decided to include the most recent version of the 6510+ Assembler on the disk. Now that can't be bad as even the cheapest of Assemblers on the market will set you back quite a few crinkly ones - so, well done, Paul.

For those among you who have never used an Assembler before I shall briefly run through the essence of Assemblers in general. During the course of lessons, however, I shall be using the 6510+ so we will learn it's particular traits together.

Sometimes mnemonics (memory joggers) are also called opcodes (operation codes). There are also other mnemonics which are referred to as pseudo-ops (pseudo operation codes). These are directives to the Assembler program itself giving it instructions, such as END (reached the end of the source code), LNK (link or chain together source code files), BYT (a directive to use a memory location for data).

The Assembler also allows us to use labels for variables, constants, memory addresses, jumps, calls or branches. A label can easily be moved and the Assembler will automatically change all those instructions which use the labels. They save figuring out memory addresses, a tediously difficult and time consuming job - especially with branch instructions (as we shall come to understand).

You are also able to add comments to your source code for better program documentation making it easier to read and use both for yourself and other programmers who may, at some time or the other, work on your program source code.

Let us now look at a part of a 'dummy' source code program which will illustrate many of the foregoing points. First, let us note the use of fields. Most Assemblers will provide you with at least four fields, namely, LABEL FIELD, OPCODE FIELD, OPERAND, or ADDRESS FIELD, and finally the COMMENT FIELD.

```
line   label      Opcode  Operand   Comment Field
No     Field      Field   Field

100              *=$C000  ;CODE START ADDRESS
110        SPRITE = $D000  ;SPRITE BASE ADDRESS
120        SCREEN = 1024   ;SCREEN BASE ADDRESS
140 START        JSR   SETUP  ;INITIAL PROGRAM SETUP
160 MAINLOOP     JSR   MOVESPRT
170              JSR   TESTSPRT
180              JSR   COLLISN
190              JSR   EXPLODE
200              JMP   MAINLOOP ;JUMP TO LABEL,
                                MAINLOOP
220 MAINPROG NOP
240 TESTSPRT     RTS   ,TEST SPRITE POSITION
250 MOVESPRT     RTS   ,MOVE SPRITE TO NEW
                       LOCATION
260 COLLISN      RTS   ;SPRITE TO SPRITE
                       COLLISION
270 EXPLODE      RTS   ;DO EXPLOSION
                        ANIMATION
290 SETUP        RTS   ;DO PROGRAM
                        SETUP HERE
310 DATA         BYT       0,1,2,3,4,5,6,7,8,9,
                           10,11,12,13,14,15
320 ADDRESS      ADR       1023,52300,2024
340 END
```

Do not worry if you cannot understand all of this, just note the use of the labels.

For example, in line 110 we have assigned the address $D000 to the label named, SPRITE - this means that anywhere in the program if we refer to SPRITE the Assembler will insert the address $D000. We could say, SPRITE+15 (or SPRITE+$0F) which would then refer to $D010. At line 160 we used the label name, MAINLOOP. After each of the instructions of line 160 through to 190 have been executed then at line 200 the program will jump back to line 160, MAINLOOP. The instruction at line 160 tells the processor to jump, saving the return address (rather like gosub), to line 250 MOVESPRT. The same with lines 170 to 190. Each instruction on the line tells the program to jump to the subroutine with the same label name. Line 310, 320 and 340 are pseudo-op codes, the first two assigning data to memory bytes and the last terminating the source code.

When you have finished writing a program or routine you will then tell the Assembler to 'assemble' the source code into object code, or machine code. The Assembler will create a 'symbol table' of all the labels and assign to each item, or label, in the table the relevant addresses. It will check through all the labels and addresses to ensure that their is no duplication; it will check each line for syntax and if it finds any syntax errors it will report them; it will also check for undefined names, illegal characters (e.g. a 2 in a binary number), invalid expressions (e.g. two opcodes in a row), illegal values (usually too large), missing operand, double definitions (i.e. two different values assigned to one name), missing labels, undefined opcodes, and et cetera.

Assuming there are no errors the source code will then be assembled into object code and located in memory to start at the address defined at the source code start (in the above dummy program $C000 (49152)). This address was determined at line 100 with the instruction * = $C000. The first piece of object code is located at line 140 - JSR SETUP. This piece of code occupies three bytes of memory, namely, 1 byte for the opcode JSR, and 2 bytes for the address of SETUP, so the label START is assigned the address of $C000. If somewhere within the program we had issued the instruction JMP START (i.e. Jump to the location labelled START) the program flow would be redirected to $C000.

The next instruction is line 160 - JSR MOVESPRT, and again three bytes are used. 1 byte for the opcode, JSR, and 2 bytes for the address of the label, MOVESPRT. As this instruction follows on from START the label MAINLOOP will be assigned with the address of $C000 + 3, or $C003. Therefore in line 200 where the processor encounters the instruction, JMP MAINLOOP, the object code will translate to JMP $C003.

I appreciate that all this probably seems like heavy going but it does, I hope, demonstrate that using an Assembler is much more easy than it first appears, especially once you have grasped the meaning of the mnemonics, or opcodes and you profusely use labels and comments. Assembling source code into object (or machine) code it does have a "one to one' relationship - no other language, compiler, or interpreter can claim this.

THE MONITOR

Most Assembler packages will contain a Machine Code Monitor, the 6510+ is no exception. Simply by typing M or M shift O (letter 'oh') you will leave the assembler and move into the Monitor, typing an X will return you to the Assembler. However, once in the Monitor you are able to disassemble any section of memory and to examine the machine code directly. A good monitor (and most are) will translate the binary opcodes into mnemonics. It will also allow you to make changes directly to the code itself. You can have the option to view code in hexadecimal format which will also list ASCII characters; this is especially useful for viewing and checking tables of data. (See Examples 1. and 2.) It is from the Monitor where you will 'run' the program you have just assembled by typing G <address>, address being the 'run' address of the program (in our earlier example this would be $C000 - G 49152, or G $C000).

The Monitor will have other commands and options, such as Fill an area of memory with a specific value, Hunt for a sequence of code or ASCII, and replace it with some other code or ASCII if necessary. Transfer code from one memory block to another. The facility to Save and Load Blocks of Machine Code. Printer options. And many others.

WORDS OF COMFORT

The going will be easy and gentle to start with, with actually learning to program in Machine Language - but, like any worth-while skill, there is a lot to learn and some difficult hills to climb. However, the reward for perseverance is both great for the personal satisfaction of overcoming a difficult task, and also for the fact that, hopefully, you will be able to put your new-found skills to very lucrative use.

I know a man, who, at the age of 48, had no idea of computers at all - in fact, he had not even touched one, not even to play games. Being an artist he did, at first, have great mental blocks for anything logically orientated making the whole quest of computer literacy a Mount Everest to climb. But now, just four years on, he programs quite successfully in Basic, Machine Language, and C. Now if he can do it - so too can you.

Take heart, there is a pot of gold at the end of the computer rainbow!

TABLE 1. The 6510 Microprocessor Instruction Set

ADC Add memory to Accumulator with Carry
AND "AND" memory with Accumulator
ASL Shift left One bit (memory or accumulator)

BCC Branch on Carry Clear
BCS Branch on Carry Set
BEQ Branch on Result Zero
BNE Branch on Result not Zero
BMI Branch on Result Minus
BPL Branch on Result Plus
BVC Branch on Overflow Clear

BVS Branch on Overflow Set
BIT Test bits in Memory with Accumulator
BRK Force Break

CLD Clear Carry Flag
CLD Clear Decimal Mode
CLI Clear Interrupt Disable Bit
CLV Clear Overflow Flag
CMP Compare Memory and Accumulator
CPX Compare Memory and Index X
CPY Compare Memory and Index Y

DEC Decrement Memory by One
DEX Decrement Index X by One
DEY Decrement Index Y by One

EOR "Exclusive-Or" Memory with Accumulator

INC Increment Memory by One
INX Increment Index X by One
INY Increment Index Y by One

JMP Jump to New Location
JSR Jump to New Location (saving return address)

LDA Load Accumulator with Memory
LDX Load Index X with Memory
LDY Load Index Y with Memory
LSR Shift Right One Bit (Memory or Accumulator)

NOP No Operation

ORA "OR" Memory with Accumulator

PHA Push Accumulator onto Stack
PHP Push Processor Status onto Stack
PLA Pull Accumulator from Stack
PLP Pull Processor Status from Stack

ROL Rotate One Bit Left (Memory or Accumulator)
ROR Rotate One Bit Right (Memory or Accumulator)
RTI Return from Interrupt
RTS Return from Subroutine

SBC Subtract Memory from Accumulator with Borrow
SEC Set Carry Flag
SED Set Decimal Flag
SEI Set Interrupt Disable Status
STA Store Accumulator in Memory
STX Store Index X in Memory
STY Store Index Y in Memory

TAX Transfer Accumulator to Index X
TAY Transfer Accumulator to Index Y
TSX Transfer Stack Pointer to Accumulator
TXA Transfer Index X to Accumulator
TXS Transfer Index X to Stack Pointer
TYA Transfer Index Y to Accumulator

EXAMPLE 1. Reproduction of a Monitor Binary Listing

ADDRESS	BYTES	ASCII
2318	00 00 00 C2 94 18 53 43	B SC
2320	55 4D 2E 4D 45 4E 55 00	UM MENU
2328	00 00 42 4F 52 44 45 52	BORDER
2330	20 53 43 52 4F 4C 4C 00	SCROLL

EXAMPLE 2. Reproduction of a Monitor Code Listing

ADDRESS	ML BYTES	MNEMONIC	OPERAND
2189	85 FA	STA	$FA
218B	A9 04	LDA	#$04
218D	85 FB	STA	$FB
218F	AD EC 21	LDA	$21EC
2192	3D D0 21	AND	$21D0
2195	FO 04	BEQ	$219A

nb. All numbers are in hexadecimal.

DOWN TO BUSINESS

In the OCTOBER 1990, JANUARY 1991 issues of CDU two articles entitled NUMBERS AND BYTES were printed. If you are unacquainted with the various numbering systems (i.e. decimal, hexadecimal and binary), and how these relate to bytes I would suggest that you take a look at these articles. However, I will cover this subject, albeit more lightly, as we proceed through these lessons. (THE FINAL ARTICLE OF THE SERIES WILL BE IN THE APRIL ISSUE)

If you do not possess a copy of the "Commodore 64 Programmer's Reference Guide", might I suggest that you do invest in a copy. You will find it an invaluable addition to your library.

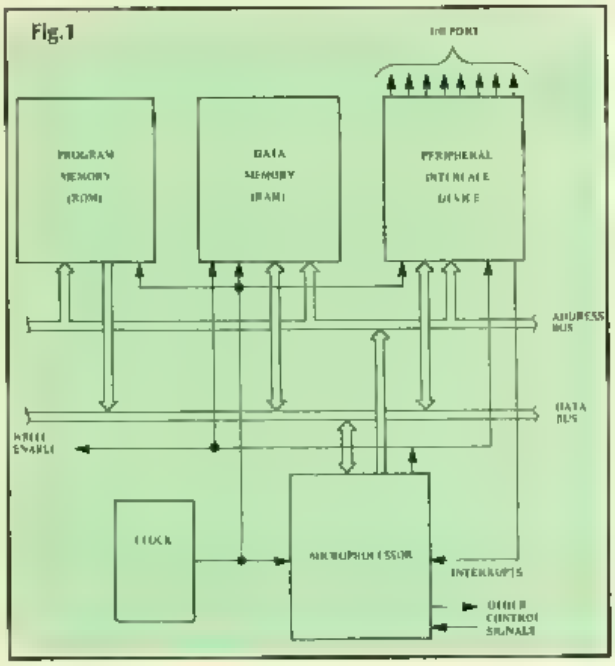
A most important aspect of programming in machine language (ML), is the need to become acquainted with the machine itself. So, before we get down to actual programming we are going to have a look at the hardware organization.

SYSTEM ARCHITECTURE

The Microprocessor Unit (MPU) (**Fig 1**), is a system which implements the functions of a Central Processing Unit (CPU) within one 'chip' (6510) (**Fig 2**). It includes an Arithmetic Logic Unit (ALU), plus its internal registers, and a Control Unit (CU), which sequences the system and decodes instructions.

The MPU creates three Buses, an eight-bit bi-directional Data bus (Db), a 16-bit three state Address bus (Ab) and a control bus. Connected to the CPU are the memory chips, giving us 65K Ram, system ROM chips, the Complex Interface Adapter (CIA), a peripheral interface device with the CPU, which has extremely flexible timing and I/O capabilities. The Sound Interface Device (SID), a single chip, 3-voice electronic music synthesizer/sound effects generator. With all of this comes the various i/o interfaces with the outside world - such as the Monitor, Keyboards, User, Cartridge, and Game ports.

Fig.1



INSIDE THE CPU

INTERNAL REGISTERS

These are bytes which are contained within the CPU, and which can be broken down into eight bits per byte. Each bit is either set (1), or clear (0). They are numbered conventionally from right to left - 7 6 5 4 3 2 1 0 - . Each bit has a value - 128 64 32 16 8 4 2 1 -. If a combination of bits in a byte are set, then we simply add together the values of each bit set, e.g. 10010011 = 128+16+2+1 = 147. 00001111 = 8+4+2+1 = 15. 01010101 = 64+16+4+1 = 85.

ACCUMULATOR REGISTER - Ac

This, the most important register in the CPU, is an eight bit register which is used to copy the contents of memory locations into the Ac, or from the Ac to memory locations. We are able to modify the contents of the Ac, and this is the only register which has instructions for performing math.

INDEX REGISTER - X

This, a very important register, functions as a counter, and, when it is used in conjunction with the Ac register, as a pointer to elements within tables. It can also be used to transfer data as the Ac but with restrictions. There are other instructions for things that only the X register can do.

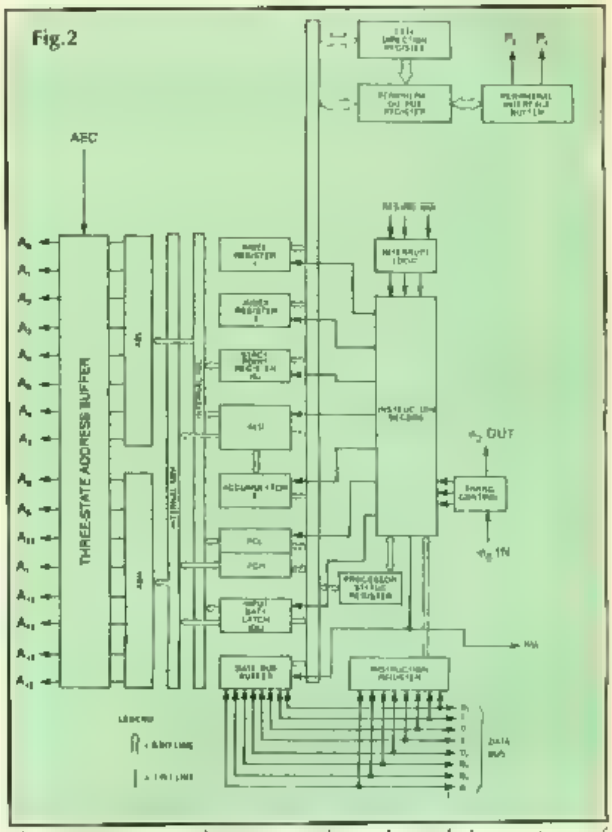
INDEX REGISTER - Y

This functions in a similar manner to the X register. There are also instructions for things that only the Y register can do.

PROGRAM COUNTER - PC

The PC is a sixteen-bit register, or word (two bytes). As the processor works its way through each instruction of the program, the PC will contain the memory location, or address, of the instruction which is being executed.

Fig.2



STACK POINTER - SP

There is an area of memory which is called the STACK (this memory is located from memory location 256 ($100) to memory location 511 ($1FF)). It is used by the system to hold important data on a temporary basis as well as return addresses when subroutines are being executed (more about the stack later). The SP is a pointer which contains the location of the first empty place on the stack.

INPUT/OUTPUT PORT - I/O PORT

Strictly this register doesn't belong in this section, however, it is used for memory management which will allow the chip to control more than 64K of ROM or RAM. Actually it is situated in RAM memory and is used in conjunction with another register (The Data Direction Register). We will cover the details of these two registers as we proceed through the series.

PROCESSOR STATUS REGISTER - SR

This register is of extreme importance to us as programmers. It is the basis for all decision making and conditional branching to other parts of the program we are creating. Of the eight bits of this register, seven of them are used as FLAGS (a flag can be viewed as something which indicates whether something has

occurred, or not occurred). A flag can be either clear, 0, or set, 1. Depending on the state of the flag a conditional branch can occur. We will see this in action very soon.

Here are the flags of the SR:

```
bit No..... ..... 7 6 5 4 3 2 1 0
flag ..., ..... NV - B D I Z C
```

Bit 0. C - CARRY. As you probably know, the value which a byte can contain is restricted to between 0 and 255. If a number greater than this is required (or less when using floating point), then we need to link bytes together. For example the first bit of the second byte will represent 256. The carry flag is used to denote an 'overflow' value from one byte to the next.

```
E.G.      byte 2      byte 1      decimal value
          00000000    11111111    255
        + 00000000    00000001    1
          --------    --------
        = 00000001    .00000000   256
              .           .
              ....carry 1...
```

Alternatively, if we are subtracting, then we can use the carry flag to indicate a borrow.

Bit 1. Z - ZERO. If the result of an operation, such as addition, subtraction, or a comparison, is zero, then this flag is set, otherwise it is clear.

```
E.G.          Ac = 10
             MEM = 10
COMPARE Ac WITH MEM result = 0 THEREFORE
ZERO FLAG SET.

              Ac = 10
             MEM =  8
COMPARE Ac WITH MEM result = 2 THEREFORE
ZERO FLAG CLEAR.
```

Bit 2. I - INTERRUPT DISABLE. We have a long way to go before we deal with interrupts, but this flag, when set, will allow us to change interrupt vectors. As I said, more on interrupts and vectors later in the series.

Bit 3. D - DECIMAL. The processor normally carries out all arithmetic in binary, however, when this flag is set, it will change the processor to decimal mode and conduct its arithmetic in decimal.

Bit 4. B - BREAK. This is not much used, except by assemblers, monitors, debuggers, and the like. The flag can indicate if execution of code was stopped by a BRK instruction. (Possibly, when we are further along in the series, we may write a routine to set a breakpoint and add it to the monitor of the 6510+, this is very useful for debugging

Bit 5. Not used.

Bit 6. V - OVERFLOW. When programming we are able to deal with signed numbers, that is positive and negative numbers. When we do use signed numbers the value a byte can contain is between -128 and +127 (binary 10000000 and 01111111 (see bit 7, below)), we need to know when we overflow that value in order to force a carry to the next higher byte, and it is here that we access the V flag.

Bit 7 N - NEGATIVE. This bit is used as the sign bit when we are dealing with signed numbers. When it is set we are dealing with negatives, and if clear then positives. You will discover that this is a very useful flag, even when not dealing with signed numbers.

ADDRESSING MEMORY.

We must have somewhere to store our data, and to fetch our data from. For this we use memory, and being a 65k computer the 64 has 65536 memory bytes available. Some of this memory is devoted to the operating system and the basic interpreter, however, as we shall see we can 'get back' this system memory.

Memory is 65536 bytes of eight bits per byte and the bytes are numbered from byte 0 to byte 65535. As you know a byte can only hold a maximum value of 255 which is obviously not large enough to access all of the 65536 memory addresses, therefore, in order to store or fetch data to or from any of the 65k locations we must have a system whereby we can access all of them. By joining two bytes together into what is conventionally termed as high byte/low byte we can now have a maximum number range of 0 to 65k - hence the PC being 16 bits wide (two bytes).

To place, or to fetch, data is termed 'memory addressing', each byte having an address in the range 0 to 65535. The memory chips are linked to the Central Processing Unit (the CPU) via electrical conductive lines on the circuit board of the computer. These are known as the Bus lines, or simply the Bus. Because we need to access an entire word (two bytes, 16 bits) to reach every address in RAM, one of the buses, the Address Bus, has 16 communicating lines linked between the CPU and Memory. However, to transfer data to or from the CPU to a Memory Address we now employ the Data Bus. Because the 64 is an eight-bit computer, this means that we are only able to transmit along the data bus a single byte at a time, it follows, therefore, that the data bus has only eight conductive lines. This is why the registers A, X, and Y are only eight bits wide, and why the PC is sixteen bits wide.

The CPU uses the data bus to transmit and to receive data in the form of electrical pulses to and from memory and to do this it utilises the Control Bus which carries the various synchronization signals which are required by the system. The CPU ensures that whilst data is being sent along the Db to memory, it does not try to receive data from memory until the bus is clear.

ARITHMETIC-LOGIC-UNIT (ALU)

The function of the ALU is to perform all arithmetic and logical operations on the data fed to it via its input ports. The ALU is equipped with a special register, the Accumulator. In arithmetic and logical operations, one

of the operands will be the Ac, and the other will be a memory location. However, the result will be deposited in the Ac. Referencing the Ac as both source and destination for data is the reason for its name: it accumulates results. This does have the advantage of being able to use short instructions, i.e. a single byte.

THE STACK

This is a structure which is formally called a LIFO (last-in, first-out) Object. Basically it is a set of RAM registers, or RAM memory locations set aside by the system and allocated as a data structure. It is chronological in its order and is 256 bytes long, and is located in Page 1 (more about paging in a moment) - that is from location 256 ($100) up to location 511 ($1FF). It is organised 'backwards' in memory, in other words, the first position in the stack is location 511 ($1FF) and the last is location 256 ($100).

The stack is used by both the system and the programmer to temporarily store data, and to remember the order of events. For example the Basic statement GOSUB must remember where it was called from so that when it meets a RETURN statement it knows where to go back to, to continue program execution. What actually occurs when a GOSUB is encountered is that the processor 'pushes' its current position (memory address) onto the stack, and then when a RETURN is encountered and executed, the processor 'pulls' off the stack the address where it must return to. Because the stack is of limited size (256 bytes) does explain why you can only 'nest' so many subroutines.

The Stack Pointer (SP) always points to the next available location on the stack. When something is 'pushed' onto the stack, the SP is decremented, and when something is 'pulled' off the stack, then the SP is incremented. As programmers we will discover the great potential of the stack - but we must also learn the dangers of its misuse.

PAGING

One Kbte = 1024. The Commodore 64 has access to 65Kbytes of RAM memory. We divide this into 'pages'. Each page is 256 ($100) bytes long. Because we count from zero, then page 0 is from 0 to 255 ($00 to $FF) locations. Page 1 is from 256 to 511 ($100 to $1FF) - the stack area. Page 3 is from 512 to 767 ($200 to $2FF), and so on. From the foregoing you will see the advantage of using hexadecimal. Whenever a page boundary is crossed, it will often introduce an extra cycle delay in the execution of an instruction. We shall discuss 'timing cycles' at a later stage.

A NOTE CONCERNING HEXADECIMAL

If you don't already have an understanding of hexadecimal it will probably seem a little hard to grasp at first, but like most things, with practise it will soon be mastered. You should try to 'think' in hex. This isn't as difficult as it may at first appear, because you won't have to think about converting a number back into decimal. For example if you said a value needs to be stored in $32FF instead of 13,055, it shouldn't make too much difference.

PROGRAMMING AT LAST

Right, now is the time to power-up your trusty 64, your disk drive, and spin the diskerooni which contains your Assembler.

We shall commence by first loading the accumulator (Ac) with the Commodore Screen Display Code for the letter "A", which has the numerical value of 1 (remember this is the Commodore screen code and not ASCII). We shall then store the content of the Ac, namely, 1, into the screen location of the top left hand corner (the screen start location, or base address being 1024 - $0400). The equivalent basic command for this would be, POKE 1024,1.

Here is the ML source code.

```
10 LDA #1
20 STA 1024
30 RTS
40 END
```

LINE 10 LDA means LoaD the Accumulator (with something). Using the Hash (#) symbol tells the assembler that we want to load the Ac with an absolute value, in our case, 1. If we omit the # symbol then the assembler would have loaded the Ac with the contents of memory location 1. The contents of memory location 1 might have been 55 ($37) so, therefore, the Ac would have been loaded with 55 ($37) and not the 1 which was required.

LINE 20 STA means STore the contents of the Accumulator, in our case 1, in the memory location which follows the STA instruction, in our small example routine, address 1024 ($0400), which is the top left screen location. We don't require the use of the # symbol here because we always store the content of the Ac to a memory location.

LINE 30 RTS means ReTurn from Subroutine. We have used this instruction to terminate the program and return back to the monitor after it has executed. If we did not use this instruction then the processor would continue stepping through memory addresses and executing anything which may follow our routine - this could be disastrous because we have no idea what values are contained in the bytes following our program. More than likely it is garbage values which could cause a system crash and in which case might require us to power off and on the computer, thereby loosing our program as a consequence, and having to reboot the assembler.

LINE 40 END is an assembler directive (pseudo-op) which tells the assembler that there is no more source code to assemble.

Now before we actually assemble this small program, we need to locate it into memory somewhere or the other, and out of the way of the assembler program itself as well as the source code and label/symbol storage areas. Usually memory location 49152 ($C000) is as good a place as anywhere. Here we have 4K of memory which we can normally use as a work space. So, having said this, let us now insert another line of code, always at the beginning of the routine, which will organise where in memory this small routine will commence from.

```
5 * = C000
```

The entire routine will now be ready for assembling. Type A shift-S (or whatever command your own assembler may use) to assemble the source code into executable object code.

Once the routine has been assembled, you can now enter the monitor and disassemble from locations $C000 (59152) to $C006. The listing will look something like this:

```
49152 C000  A9 01        LDA #$01
49154 C002  8D 00 04     STA $0400
49157 C005  60           RTS
49158 C006  00           BRK
```

The two left-hand columns are the memory addresses in decimal and hex, this is followed by hexadecimal byte values which represent the binary machine code, and the right column is a mnemonic listing with hexadecimal operands.

If you now run this small routine, i.e. type G C000, the letter A will be printed at the top left of the screen, and the routine will terminate back to the monitor and, depending upon the assembler you use, might list the state of the registers.

AN ARITHMETIC PROGRAM

Now that we have actually constructed a routine to get the feel of things let's quickly 'bash' on an construct a routine which will add together two eight-bit operands. The use of labels improves the readability of the source code, and, should we need to change a value of a memory address, this will make such a change more simple.

The 'header' of our routine will organise a memory location where the program will be assembled to, carry assembler directives, or pseudo-ops, and initialise our equates.

```
10 ; *** 8-BIT ADDITION ROUTINE ***
20 ;
30  * = $C000 ; START LOCATION FOR  ROUTINE
40  OUT       ; DIRECTIVE TO LIST OUTPUT DURING
                ASSEMBLY
60 ; *** EQUATES ***
80  ADDRESS1 = $C100  ; STORAGE FOR OPERAND 1
90  ADDRESS2 = $C101  ; STORAGE FOR OPERAND 2
100 ADDRESS3 = ADDRESS2+1 ; STORAGE FOR THE
                            RESULT
120 ; *** PROGRAM START ***
125 ;
130 SETUP  LDA OP1     ; VALUE OF OPERAND 1
140        STA ADDRESS1  ; STORE IT
150        LDA OP2     ; VALUE OF OPERAND 2
160        STA ADDRESS2  ; STORE IT
170 ;
180 MAIN  LDA ADDRESS1 ; LOAD OPERAND 1 INTO
                         AC.
190       CLC     ; CLEAR THE CARRY FLAG
200       ADC  ADDRESS2 ; ADD OPERAND 2 TO
                          OPERAND 1
210       STA ADDRESS3 ; SAVE RESULT TO ADDRESS
3
220       RTS     ; TERMINATE ROUTINE, RETURN TO
                    MONITOR
250 OP1   BYT 10  ; STORE THE VALUE 10 INTO THE
                    VARIABLE OP1
260 OP2   BYT 100     ; STORE THE VALUE 100 INTO
                        THE VARIABLE OP2
280       END         ; END ASSEMBLY
```

I've constructed a small routine here which uses the comment field to demonstrate a well documented program. Should this be put aside for a long while and then returned to it will be relatively easy to refresh the mind exactly what the routine is actually being used for. Naturally this is a simple demonstration, however, it serves well as a demo' for when routines become much more complex and vital information may be forgotten.

UNFORTUNATELY, DUE TO LACK OF SPACE IN THIS MONTHS ISSUE, THE LINE BY LINE EXPLANATION OF THE ABOVE PROGRAM WILL HAVE TO BE LEFT OUT. AT THE START OF THE NEXT INSTALLMENT WE WILL INCLUDE THE BREAKDOWN OF THIS PROGRAM FOR YOUR PERUSAL. WE APOLOGISE TO ALL THOSE THAT WILL MISS THIS EXPLANATION.

IN CONCLUSION

That's all for this month. Most of this lesson has been involved in explaining how the system works, and we actually only just managed to get down to an actual bit of programming. You will find that understanding the system will aid you greatly as your programming expertise increases. However we have covered the method to load and store the accumulator, probably the most used and definitely the most important register in the set. We examined the use of the ADC method of adding together two eight-bit values and the importance of clearing the carry bit, CLC. You can experiment with what we have covered in this issue to you hearts content until next issue where we conduct a sixteen-bit addition, some subtraction and then we will enter the world of loops, and branches, which will lead us into multiplication and binary shifts.

Till then take care...

Lineage: 58p per word +VAT)
Semi display £11.50 plus VAT per single column centimetre minimum 2cm. Ring for information on series bookings/discounts

All advertisements in this section must be prepaid
Advertisements are accepted subject to the terms and conditions printed on the advertisement rate card (available on request)

0908 569819

Send your requirements to:
CDU CLASSIFIED DEPARTMENT, ALPHAVITE PUBLICATIONS LTD ,
20 POTTERS LANE, KILN FARM, MILTON KEYNES, MK11 3HF.

SOFTWARE

SILVER WING SOFTWARE

THE DEDICATED PD LIBRARY

IN OUR SECOND YEAR OF SUPPLYING 64 PD, WE STILL CONTINUE TO PROVIDE THE MOST USER FRIENDLY SERVICE AROUND! WE HAVE A WIDE SELECTION OF ALL THE LATEST MIND BLOWING DEMOS THAT REALLY SHOW OFF JUST WHAT THE CBM 64 IS CAPABLE OF DOING EG CIANT SPRITE MULTI-PLEXORS, BOBS, VECTORS PLOTTERS, SINUSES, DYSP'S, DYPP S & MANY MORE COMPLEX ROUTINES!

MANY DEMOS ARE SO IMPRESSIVE THEY WOULON T EVEN LOOK OUT OF PLACE ON THE AMIGA! EACH ONE IS CREATED BY THE WORLDS BEST DEMO GROUPS & IS PROGRAMMED TO A HIGH STANDARD WITH GREAT GRAPHICS & ORIGINAL MUSIC ALSO INCLUDED WE ALSO HAVE AN EXCELLENT COLLECTION OF THE BEST PD UTILITIES, MANY OF WICH ARE BETTER THAN COMMERCIALLY RELEASED ONES! THEY INCLUDE GRAPHICS SFX & MUSIC EDITORS DISK COPIERS, CRUNCHERS LETTER WRITERS & MANY MANY MORE WE EVEN HAVE A FEW PROGRAMMES THAT WILL LET A COMPLETE NOVICE CREATE A DEMO OF THEIR OWN! ALL UTILITIES ARE OF HIGH QUALITY & REALLY HAVE TO BE SEEN TO BELIVED IF YOU ENJOY A QUICK GAME OR TWO THEN LOOK NO FURTHER WE HAVE A RANGE OF THE BEST PD GAMES AROUND ARCADE, PUZZLE & ADVENTURE SOME OF WICH ARE COMMERCIAL QUALITY! WE RE NOT ONLY THE LEADING SUPPLIER OF 64PD BUT WE ALSO OFFER FREE EXPERT ASSISTANCE SHOULD ANY PROGRAMMING OR PD PROBLEMS ARISE

FOR A COPY OF OUR LATEST LIST SEND AN SAE TO THE FOLLOWING ADDRESS, ALTERNATIVELY SEND £2.00 FOR A MORE DETAILED COMPUTERISED LIST & A FEW EXTRA DEMOS.

SILVER WING SOFTWARE
185 CALLOWBROOK LANE
RUBERY,
BIRMINGHAM B45 9TG

CBM 64 PUBLIC DOMAIN

CLASSIFIED COUPON

ALL CLASSIFIED ADVERTISEMENTS MUST BE PRE-PAID.
THERE ARE NO REIMBURSEMENTS FOR CANCELLATIONS.

CDU CLASSIFIED DEPARTMENT, ALPHAVITE PUBLICATIONS,
20 POTTERS LANE, KILN FARM, MILTON KEYNES, MK11 3HF.

RATES Lineage 58p per word (+VAT). Semi-display: £11.50 (+VAT) per single column cm minimum size 2cm Series discounts available

I enclose my Cheque/Postal Order for £ for insertions, made payable to Alphavite Publications
(Delete as necessary)

PLEASE DEBIT MY ACCESS/VISA

CARD NO.:

EXP DATE:

£ FOR INSERTIONS

Name:

Address:

Post Code..

Daytime Tel No.:

Signature: Date:

☐ FOR SALE ☐ SOFTWARE ☐ SPECIAL OFFERS ☐ OTHER

ONLY POOLS AND HORSES

Every program written by a mathematician who has spent many years in the betting industry. Programs that utilise the tried and trusted methods of the professional, not pie in the sky theories that fail to pass the test of time.

FOOTBALL BOXFORM Written by a former pools expert for Littlewoods. The program has forecast over 50% more draws than would be expected by chance
Homes, Aways and draws shown in order of merit and true odds given for every match Merit tables show at a glance the teams currently in form and those currently having a lean spell. Australian pools program included in the price.

POOLS PLANNER by the same author. Full details given of 369 easily entered block perms ranging from 9 to 73960 lines and from 12 to 56 selections. All are accepted by the pools firms and are checked in seconds by your computer.

RACING BOXFORM Course characteristics (built in to the program) as well as the form of the horses are considered to speedily produce an order of merit for each race Designed for flexibility allowing users to amend the program if they wish. Price still includes the highly acclaimed HANDI CAP WINNER - more than 1000 winners every year - over 25% of them at 5/1 or better Order two or more and recieve **FREE** a program to work out almost any bet. So good its used by bookies.

Prices (Tape) £15.95 each. £25.95 any two. £35.95 all three. For discs please add £2. per program.

Advertised for Six years in the sporting and computing press.
BOXOFT CLEVER.. GET THE BEST

BOXOFT (CDU), 65 Allans Meadow,
Neston, South Wirral L64 9SQ
Cheque/ P.O./Access/Visa Tel:051-336-2668

VISA

BBC B & MASTER, AMSTRAD CPC & PCW COMMODORE 64/128, SPECTRUMS.

DEALERS

WORTHING COMPUTERS

Tel: 0903 210861

**7 WARWICK ST
WORTHING
WEST SUSSEX**

Stockists of a large range of 8 Bit and 16 Bit Software and Peripherals.

REPAIRS

REPAIRS

C64, C+4 .... £30 inc
C128 1541.. .....£40 inc.
Three month warranty 1 week turnround

SPARES

906114	£9.30	8501	£11.75
6510	£11.50	C64 Power Pack	£22.00
6526	£11.95	C64 Cassette unit	£22.00
6581	£14.95		
6569	£21.50		

OASIS COMPUTER SERVICES
Dept 4C 14 Ridgeway Road,
Salisbury, Wiltshire, SP1 3BU
Tel:(0722) 336061

**COMMODORE 64/128
SOFTWARE LIBRARY**

- Life Membership
- Updates
- Teenage Mutant Hero Turtle Tape £1.75
- 7 Day Hire
- Tapes From £1.50 to £1.75
- Also Disks.

FOR FREE CATALOGUE S.A.E TO –
ACE SOFTWARE LIBRARY
14 CHIPENDALE COURT BELFAST, BT10 0DU

...it's dynamite!

POWER CARTRIDGE

FOR YOUR COMMODORE 64/128

"...unbelievable value for money" ZZAPP! Dec 89

- POWER TOOLKIT
- POWER MONITOR
- TAPE & DISK TURBO
- PRINTERTOOL
- POWER RESET
- TOTAL BACKUP

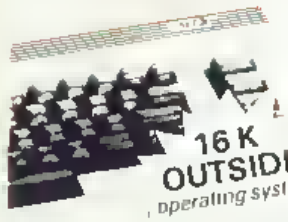
..."Money well spent" YC/CDU Jan 90

42 Pg Manual - "Damned Good Handbook" CCI Jan 90

SO MUCH FOR SO LITTLE

AVAILABLE FROM ALL GOOD COMPUTER RETAILERS

TRIED AND TESTED - OVER 100,000 SOLD IN EUROPE



...."highly recommended for C64 users" CCI Jan 90

YOU WILL WONDER HOW YOU EVER MANAGED WITHOUT IT

[illegible]

A powerful BASIC toolkit (Additional helpful commands) that tremendously simplifies programming and debugging.

AUTO	HARDCAT	RENUMBER
AUDIO	HARDCOPY	REPEAT
COLOR	HEX$	SAVE
DELETE	INFO	TRACE
DELETE	KEY	UNNEW
DOKE	PAUSE	QUIT
DUMP	PLIST	MONITOR
FIND	RLOAD	RSAVE

RENUMBER *Also modifies all the GOTO's GOSUB's etc. Allows part of a program to be renumbered or displaced*

PSET *Set up of printer type*

HARDCAT *Prints out Directory*

The toolkit commands can be used in your programs

[illegible]

Using POWER CARTRIDGE you can load up to 8 times faster from disk. The Disk commands can be used in your own programs.

DLOAD	DVERIFY	DIR
DSAVE	MERGE	DEVICE
DISK		

MERGE *Two BASIC programs can be merged into one.*

DISK *With DISK you can send commands directly to your disk.*

[illegible]

Using POWER CARTRIDGE you can load up to 8 times faster from disk with your data recorder. The tape commands can be used in your own programs.

TLOAD	TSAVE	TVERIFY
MERGE	AUDIO	

[illegible]

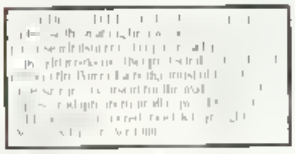
A powerful machine language monitor that is readily available and leaves all of your 64 commodore memory available for programming.
Also works in BASIC ROM, RAM and I/O areas.

A ASSEMBLE	I INTERPRET	S SAVE
C COMPARE	J JUMP	T TRANSFER
D DISASSEMBLE	L LOAD	V VERIFY
M MEMORY	W WALK	
H HUNT	P PRINT	X EXIT
G GO	R REGISTER	$ DIRECTORY DOS Commands

[illegible]

The POWER CARTRIDGE contains a very effective Printer Interface that will detect if a printer is connected to the Serial Bus or User Port.
It will print all Commodore characters on Epson and compatible printers.
The printer interface has a variety of set up permissions. It can produce HARDCOPY of screen not only on Serial printer (MPS801, 802, 803 etc.) but also on Centronics printers (EPSON, STAR, CITIZEN, PANASONIC etc.)
The HARDCOPY function recognises the difference between HIRES and LORES. Multi-colour graphics are converted into shades of grey. The PSET function allows you to decide on large/small and Normal/inverse printing.
The printer PSET function can:

PSET 0	*Set date (via Serial Interface)*
PSET 1	*EPSON mode only*
PSET 2	*SWITCH CITIZEN mode only*
PSET 3	*Turn the printing 90 degrees*
PSET 4	*HARDCOPY setting for MPS801/1526*
PSET 8	*Bit image mode*
PSET C	*Setting Lower/Upper case and sending Linefeed codes*
PSET T	*All characters are printed in an unmodified mode*
PSET U	*Runs a Serial printer and leaves the User port available*
PSET Sx	*Sets the secondary address for HARDCOPY with Serial Bus*
PSET II	*Adds a line-feed (LF) after each line*
PSET LO	*Switches PSET II off*



[illegible]

On the back of the POWER CARTRIDGE there is a Reset Button. Pressing this button makes a SPECIAL MENU appear on the screen.
This function will work with many programmes.

CONTINUE	*Allows you to return to your program*
BASIC	*Returns to BASIC*
RESET	*Normal RESET*
TOTAL	*Erases the contents of the memory, even a Back Up program can be activated (ink with RELOAD if there is no LOADING)*
BACKUP	*RESETs any program as BACKUP ENTER but no TAPE*
DISK	
TAPE	
HARDCOPY	*Starts monitor, prints out a Hardcopy of the screen. Using CONTINUE may overwrite any data output to the printer*
MONITOR	*Enters you into the Machine language Monitor*

BDL

Bitcon Devices Ltd

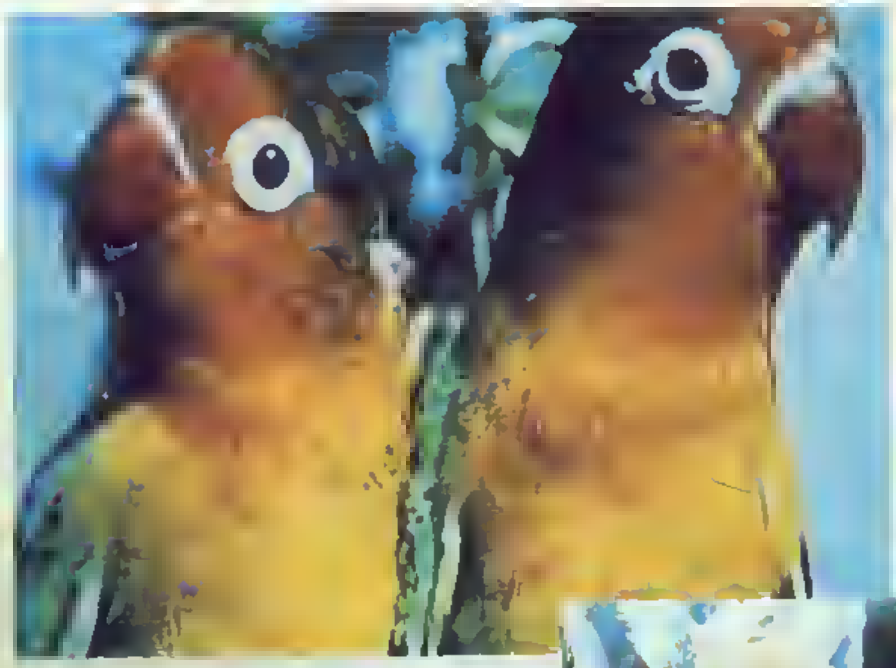
88 BEWICK ROAD
GATESHEAD
TYNE AND WEAR
NE8 1RS
ENGLAND

Tel 091 490 1975 and 490 1919 Fax 091 490 1918
To order: Access/Visa welcome - Cheques or P/O payable to BDL
Price £16.99 incl. VAT.
UK orders add £1.20 post/pack total - £18.19 incl. VAT.
Europe orders add £2.50, Overseas add £3.50
Scandinavian Mail Orders and Trade enquiries to: Bhak Elektronik, Box 216, Norrtalje 76123, SWEDEN. Tel: +46 176 18425 Fax: +76 18401
TRADE AND EXPORT ENQUIRIES WELCOME

THE COMPLETE COLOUR SOLUTION

£179

Vidi ... No 1 in UK & Europe (Leading the way forward)



Get the most out of your Amiga by adding:

"The Complete Colour Solution"

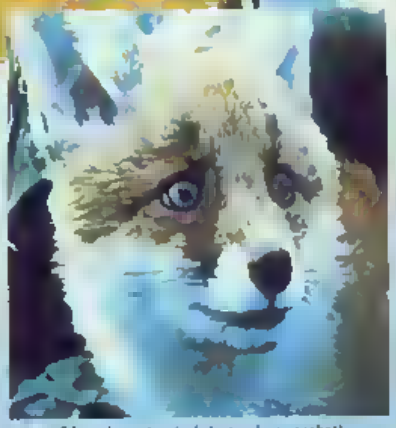
The Worlds ultimate creative leisure product for your Amiga. Capture dynamic high resolution images into your Amiga in less than one second.

And Look No Filters

Images can now be grabbed from either colour video camera, home VCR or in fact any still video source. The traditional method of holding three colour filters in front of your video camera is certainly a thing of the past. Because Vidi splits the RGB colours electronically there are no focussing or movement problems experienced by some of our slower competitors. Lighting is also less of an issue as light is not being shut out by lens filters. Put all this together with an already proven Vidi-Amiga/VidiChrome combination and achieve what is probably the most consistent and accurate high quality 4096 colour images ever seen on the Amiga.

The colour solution is fully compatible with all Amiga's from a standard A500 to the ultimate A3000. No additional RAM is required to get up and running.

You will see from independant review comments that we are undoubtedly their first choice and that was before the complete solution was launched. If you have just purchased your Amiga and are not sure what to buy next, then just read the comments or send for full review and demo disk.



"Actual unretouched digitised screenshot"

Features ...

- Grab mono images from any video source
- Capture colour images from any still video source
- Digitise up to 16 mono frames on a 1meg Amiga
- Animate 16 shade images at different speeds
- Create windows in both mono & colour
- Cut & Paste areas from one frame to another
- Hardware and software brightness & contrast control
- Choice of capture resolutions standard & Dynamic Interlace
- Full Palette control
- Add text or draw within art package

Amiga Computing: The best Amiga digitiser has had the technicolour treatment. Vidi must be one of the most exciting peripherals you can buy for your Amiga.

Music Mart: When I first saw Vidi "in the flesh" as it were at the CES show last September it looked to be the answer to a frustrated Digi View owner's dreams – in fact to see pictures appearing on screen without the customary two minues wait seemed almost too good to be true. I have consistantly produced more good quality pictures in the short time I have had Vidi than I ever did with Digiview.

Zoom: Now under normal circumstances cheap usually means poor quality but this is not the case with Rombo. Why? cos Vidi-Amiga is the best digitiser for under £500 and I've tried them all.

Amiga Format: Where quality is concerned Vidi produces some of the best results I've seen on any digitiser at any price.

The latest addition to the Rombokit is called Vidi-RGB and brings this already impressive package to the realms of totally amazing CONCLUSION. Who will find Vidi Amiga useful? The answer to this is almost anyone with a video recorder or camera and a passing interest in graphics.

*Full colour demonstration [illegible] available for only £1.95 – cover P&P **

6 Fairbairn Road, Livingston, EH54 6TS. Tel: 0506-414631 Fax: 0506-414634

ROMBO Limited